aws

Benutzerhandbuch

AWS Glue

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Glue: Benutzerhandbuch

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist AWS Glue? 1
AWS Glue features 2
Erfahren Sie mehr über Innovationen in AWS Glue 4
Erste Schritte mit AWS Glue 4
Zugriff AWS Glue 4
Zugehörige Services 5
Funktionsweise 6
Isolierte, Serverless-ETL-Aufträge 7
Konzepte 8
AWS Glue Terminologie 11
Komponenten 14
AWS Glue Konsole 15
AWS Glue Data Catalog 15
AWS Glue Crawler und Klassifikatoren 16
AWS Glue ETL-Operationen 16
ETL einstreamen AWS Glue 17
Das Jobsystem AWS Glue 17
Visuelle ETL-Komponenten 18
AWS Glue für Spark und AWS Glue für Ray 24
Was ist AWS Glue für Ray? 25
Umwandeln semistrukturierter Schemas in relationale Schemas 26
AWS Klebertypen 28
AWS Typen von Glue-Datenkatalogen 28
Typen in AWS Glue mit Spark-Skripten 28
AWS Typen von Glue Crawler 29
Erste Schritte 30
Überblick über die Verwendung AWS Glue 30
Einrichten von IAM-Berechtigungen 32
Nächste Schritte 39
Einrichtung für AWS Glue Studio 40
Erste Schritte mit Notebooks in AWS Glue Studio 53
Nutzungsprofile einrichten 56
Nutzungsprofile verwalten 58
Nutzungsprofile und Jobs 70

Erste Schritte mit dem AWS Glue Data Catalog ... 71
Übersicht ... 71
Schritt 1: Erstellen einer Datenbank ... 71
Schritt 2. Erstellen einer Tabelle ... 73
Nächste Schritte ... 74
Netzwerkzugriff auf Datenspeicher einrichten ... 78
Einrichten einer VPC zum Herstellen einer Verbindung mit PyPI für AWS Glue ... 79
Einrichten des DNS in Ihrer VPC ... 81
Einrichten der Verschlüsselung ... 82
Netzwerke für die Entwicklung einrichten ... 87
Einrichten Ihres Netzwerks für einen Entwicklungsendpunkt ... 87
Amazon EC2 für einen Notebook-Server einrichten ... 89
Datenermittlung und Katalogisierung ... 91
Den Datenkatalog auffüllen ... 94
Verwenden eines AWS-Glue-Crawler ... 95
Manuelles Definieren von Metadaten ... 187
Integration mit anderen AWS Diensten ... 208
Einstellungen für den Datenkatalog ... 209
Auffüllen und Verwalten von Transaktionstabellen ... 212
Erstellen von Iceberg-Tabellen ... 213
Optimieren von Iceberg-Tabellen ... 216
Optimierung der Abfrageleistung für Iceberg-Tabellen ... 250
Verwaltung des Datenkatalogs ... 252
Aktualisierung des Schemas und Hinzufügen neuer Partitionen ... 253
Optimieren der Abfrageleistung mithilfe von Spaltenstatistiken ... 261
Verschlüsseln Ihres Data Catalog ... 287
Sicherung Ihres Datenkatalogs mit Lake Formation ... 287
Mit AWS Glue Data Catalog Ansichten arbeiten ... 288
Zugriff auf den Datenkatalog ... 292
Mit dem AWS Glue Iceberg-REST-Endpunkt eine Verbindung zum Datenkatalog herstellen ... 294
Verbindung zum Datenkatalog mithilfe des AWS Glue Iceberg-REST-Erweiterungsendpunkts herstellen ... 297
AWS Glue REST APIs für Apache Iceberg ... 298
Von einer eigenständigen Spark-Anwendung aus eine Verbindung zum Datenkatalog herstellen ... 318

Datenzuordnung zwischen Amazon Redshift und Apache Iceberg 319
Überlegungen und Einschränkungen bei der Verwendung von AWS Glue Iceberg REST Catalog APIs 320
Bewährte Methoden für den Datenkatalog 321
Überwachung der Nutzungsmetriken des Datenkatalogs in Amazon CloudWatch 322
Überblick über die Metriken von Data Catalog 322
Hinzufügen von Metriken zu Ihrem Dashboard CloudWatch 323
..... 324
AWS Glue Schema Registry 325
Schemata 326
Registrierungen 328
Schema-Versioning und -Kompatibilität 329
Open-Source-SerDe-Bibliotheken 335
Kontingente in Schema Registry 335
Funktionsweise 336
Erste Schritte 338
Integration in AWS Glue Schema Registry 361
Migration zu AWS Glue Schema Registry 388
Herstellen einer Verbindung zu Daten 391
Übersicht zur Verwendung von Connectors und Verbindungen 391
Vereinheitlichte Verbindungen 392
Unterstützte Authentifizierungstypen 396
Überlegungen 397
Verfügbare Verbindungen 397
Einschränkungen 401
AWS Glue Verbindungseigenschaften 401
Erforderliche Verbindungseigenschaften 402
Eigenschaften der JDBC-Verbindung 403
Verbindungseigenschaften von MongoDB und MongoDB Atlas 408
Eigenschaften der Salesforce-Verbindung 409
Snowflake-Verbindung 410
Vertica-Verbindung 411
SAP-HANA-Verbindung 412
Azure-SQL-Verbindung 413
Teradata-Vantage-Verbindung 414
OpenSearch Verbindung zum Dienst 415

Azure-Cosmos-Verbindung 416
SSL-Verbindungseigenschaften 416
Kafka-Verbindungseigenschaften für die Authentifizierung 419
BigQuery Google-Verbindung 420
Vertica-Verbindung 411
Speichern von Verbindungsanmeldedaten in AWS Secrets Manager 421
AWS Glue Verbindung hinzufügen 422
Verbindung zu Adobe Analytics herstellen 424
Verbindung zu Adobe Marketo Engage herstellen 433
Verbinden mit Amazon Redshift 443
Verbindung zu Asana herstellen 448
Verbindung zu Azure Cosmos DB herstellen 458
Verbindung zu Azure SQL herstellen 462
Verbindung zu Blackbaud Raiser's Edge NXT herstellen 465
Verbindung zu CircleCI herstellen 476
Verbindung zu Datadog herstellen 488
Verbindung zu Docusign Monitor herstellen 496
Verbindung zu Domo herstellen 504
Verbindung zu Dynatrace herstellen 513
Verbindung zu Facebook-Anzeigen herstellen 519
Verbindung zu Facebook Page Insights herstellen 527
Verbindung zu Freshdesk herstellen 551
Verbindung zu Freshsales herstellen 561
Verbindung zu Google Ads herstellen 568
Verbindung zu Google Analytics herstellen 4 597
Verbindung zu Google herstellen BigQuery 610
Verbindung zur Google Search Console herstellen 615
Verbindung zu Google Sheets herstellen 625
Verbindung herstellen zu HubSpot 632
Verbindung zu Instagram-Anzeigen herstellen 660
Verbindung zu Intercom herstellen 668
Verbindung zu Jira Cloud herstellen 690
Verbindung zu Kustomer herstellen 717
Verbindung herstellen zu LinkedIn 746
Verbindung zu Mailchimp herstellen 754
Verbindung zu Microsoft Dynamics 365 CRM herstellen 765

Verbindung zu Microsoft Teams herstellen 773
Verbindung zu Mixpanel herstellen 784
Verbindung zum Montag herstellen 800
Verbindung zu MongoDB herstellen 810
Verbindung zu Oracle herstellen NetSuite 814
Verbindung zum OpenSearch Service herstellen 826
Verbindung zu Okta herstellen 830
Verbindung herstellen zu PayPal 839
Verbindung zu Pendo herstellen 847
Verbindung zu Pipedrive herstellen 856
Verbindung zum Productboard herstellen 865
Verbindung herstellen zu QuickBooks 874
Verbindung zu Salesforce herstellen 882
Verbindung zur Salesforce Marketing Cloud herstellen 895
Verbindung zur Salesforce Commerce Cloud herstellen 917
Verbindung zu Salesforce Marketing Cloud Account Engagement herstellen 928
Verbindung zu SAP HANA herstellen 942
Verbindung zu SAP herstellen OData 946
Verbindung herstellen zu SendGrid 969
Verbindung herstellen zu ServiceNow 979
Verbindung zu Slack herstellen 991
Verbindung zu Smartsheet herstellen 1003
Verbindung zu Snapchat-Anzeigen herstellen 1015
Herstellen einer Verbindung zu Snowflake 1025
Verbindung zu Stripe herstellen 1029
Verbindung zu Teradata herstellen 1080
Verbindung zu Twilio herstellen 1084
Verbindung zu Vertica herstellen 1104
Verbindung herstellen zu WooCommerce 1108
Verbindung zu Zendesk herstellen 1138
Verbindung zu Zoho CRM herstellen 1168
Verbindung zu Zoom-Meetings herstellen 1179
Hinzufügen einer JDBC-Verbindung mit Ihren eigenen JDBC-Treibern 1190
Verwenden von benutzerdefinierten Konnektoren und Verbindungen 1195
Erstellen von benutzerdefinierten Connectors 1197
Erstellen von Verbindungen für Connectors 1202

Erstellen von Aufträgen mit benutzerdefinierten Connectors 1210
Verwalten von Connectors und Verbindungen 1218
Entwickeln von benutzerdefinierten Connectors 1221
Einschränkungen für die Verwendung von Anschlüssen und Verbindungen in AWS Glue Studio 1224
Testen eines AWS Glue Verbindung 1224
Konfiguration von AWS Anrufen für die Weiterleitung über Ihre VPC 1225
Herstellen einer Verbindung mit einem JDBC-Datenspeicher in einer VPC 1226
Zugriff auf VPC-Daten über Elastic Network-Schnittstellen 1227
Eigenschaften der Elastic Network-Schnittstelle 1228
Verwenden einer MongoDB- oder MongoDB-Atlas-Verbindung 1228
Crawling eines Amazon-S3-Datenspeichers mit einem VPC-Endpunkt 1229
Voraussetzungen 1230
Herstellen der Verbindung zu Amazon S3 1231
Testen der Verbindung zu Amazon S3 1234
Erstellen eines Crawlers für einen Amazon-S3-Datenspeicher 1236
Erstellen eines Crawlers für Amazon-S3-unterstützte Datenkatalog-Tabellen 1238
Ausführen eines Crawlers 1239
Fehlerbehebung 1239
Arbeiten mit AWS Lake Formation-geschützten Daten 1240
Vollständiger Tabellenzugriff 1240
Fehlerbehebung bei Verbindungsproblemen 1241
Tutorial: Den AWS Glue Connector für Elasticsearch verwenden 1242
Voraussetzungen 1243
Schritt 1: (Optional) Erstellen Sie einen AWS geheimen Schlüssel für Ihre OpenSearch Clusterinformationen 1243
Schritt 2: Abonnieren des Konnektors 1244
Schritt 3: Aktivieren Sie den Connector AWS Glue Studio und stellen Sie eine Verbindung her 1245
Schritt 4: Konfigurieren einer IAM-Rolle für Ihren ETL-Auftrag 1246
Schritt 5: Erstellen Sie einen Job, der die OpenSearch Verbindung verwendet 1247
Schritt 6: Ausführen des Auftrags 1248
Schaffung von AWS Glue Arbeitsplätzen mit interaktiven Sitzungen 1250
Überblick über AWS Glue interaktive Sessions 1250
Einschränkungen 1251
Erste Schritte mit AWS Glue interaktive Sitzungen 1251

Voraussetzungen für die lokale Einrichtung von interaktiven Sitzungen 1251
Installation von Jupyter und AWS Glue interaktiven Sitzungen (Jupyter-Kernel) 1251
Ausführen von Jupyter 1252
Konfigurieren der Anmeldeinformationen und der Region für die Sitzung 1252
Aktualisieren aus der Vorschau der interaktiven Sitzungen 1254
Interaktive Sitzungen mit SageMaker AI Studio verwenden 1254
Verwenden von interaktiven Sitzungen mit Microsoft Visual Studio Code 1254
Interaktive Sitzungen mit IAM 1258
Konfigurieren von AWS Glue-interaktiven Sitzungen für Jupyter und AWS Glue Studio-Notebooks 1267
Einführung in Jupyter Magics 1267
Unterstützte Magics in AWS Glue-interaktiven Sitzungen für Jupyter 1267
Benennen von Sitzungen 1286
Angeben einer IAM-Rolle für interaktive Sitzungen 1287
Konfigurieren von Sitzungen mit benannten Profilen 1288
Ein Skript oder ein Notizbuch in einen AWS Glue Job umwandeln 1289
Arbeiten mit Streaming-Vorgängen in AWS Glue interaktive Sitzungen 1289
Wechseln des Streaming-Sitzungstyps 1289
Sampling des Eingabestreams für die interaktive Entwicklung 1289
Ausführen von Streaming-Anwendungen in interaktive Sitzungen 1291
Lokal entwickeln und testen 1292
Entwickeln mit AWS Glue Studio 1293
Entwickeln mit interaktiven Sitzungen 1293
Lokale Entwicklung von AWS Glue Jobs mit Docker 1293
Entwicklungsendpunkte 1304
Migrieren von Entwicklungsendpunkten zu interaktiven Sitzungen 1306
Entwickeln von Skripts unter Verwendung von Entwicklungsendpunkten 1307
Verwalten von Notebooks 1337
Erstellung visueller ETL-Jobs 1339
Erstellen Sie visuelle ETL-Jobs mit AWS Glue Studio 1339
Anmelden in der Konsole 1339
Nächste Schritte zum Erstellen eines Auftrags in AWS Glue Studio 1340
Erstellen Sie visuelle ETL-Flows mit Amazon SageMaker 1340
Visuelle ETL-Jobs werden gestartet in AWS Glue Studio 1340
Jobs starten in AWS Glue Studio 1341
Features des Auftragseditors 1343

Transformieren Sie Daten mit AWS Glue verwaltete Transformationen 1351
Transformieren Sie Daten mit benutzerdefinierten visuellen Transformationen 1431
Verwenden von Data Lake-Frameworks mit AWS Glue Studio 1449
Herstellen von Verbindungen mit Datenquellen 1461
Konfigurieren von Datenzielknoten 1471
Bearbeiten oder Hochladen eines Auftragsskripts 1477
Ändern der übergeordneten Knoten für einen Knoten im Auftragsdiagramm 1481
Löschen von Knoten aus dem Auftragsdiagramm 1482
Hinzufügen von Quell- und Zielparametern zum AWS Glue Knoten Datenkatalog 1486
Verwendung von Git-Versionskontrollsystemen in AWS Glue 1488
Code erstellen mit AWS Glue Studio notebooks 1497
Einschränkungen 1497
Übersicht über die Verwendung von Notebooks 1498
Erstellen eines ETL-Jobs mithilfe von Notebooks in AWS Glue Studio 1499
Notebook-Editor-Komponenten 1500
Speichern von Notebook und Auftragsskript 1501
Verwalten von Notebooksitzungen 1502
Amazon Q Developer mit AWS Glue Studio Notebooks verwenden 1504
Überwachen von Jobläufen 1504
Zugriff auf das Dashboard für die Auftragsüberwachung 1505
Übersicht über das Dashboard zur Auftragsüberwachung 1505
Anzeigen von Auftragsausführungen 1505
Anzeigen der Auftragsausführungsprotokolle 1510
Anzeigen der Details einer Auftragsausführung 1511
Amazon CloudWatch Metriken für eine Spark-Jobausführung anzeigen 1514
Amazon CloudWatch Metriken für einen Ray-Joblauf anzeigen 1515
Erkennen und Verarbeiten von sensiblen Daten 1517
Auswahl der Scan-Methode der Daten 1518
Auswählen der zu erkennenden PII-Entitäten 1519
Angeben der Erkennungsempfindlichkeit 1523
Auswahl, was mit identifizierten PII-Daten zu tun ist 1524
Hinzufügen detaillierter Aktionsüberschreibungen 1525
Verwalten von Aufträgen 1526
Starten einer Auftragsausführung 1526
Planen von Auftragsausführungen 1527
Verwalten von Auftragsplänen 1528

Anhalten von Auftragsausführungen ... 1529
Anzeigen von Aufträgen ... 1530
Anzeigen von Informationen zu den letzten Auftragsausführungen ... 1530
Anzeigen des Auftragsskripts ... 1532
Ändern der Auftragseigenschaften ... 1532
Speichern des Auftrags ... 1535
Klonen von Aufträgen ... 1538
Löschen von Aufträgen ... 1538
Arbeiten mit Aufträgen ... 1540
AWS Glue Versionen ... 1540
AWS Glue Versionen ... 1540
AWS Glue Richtlinie zur Versionsunterstützung ... 1560
Migration AWS Glue für Spark-Jobs auf AWS Glue Version 5.0 ... 1562
Migration AWS Glue für Spark-Jobs auf AWS Glue Version 4.0 ... 1577
Aktualisieren Sie die Analyse mit KI ... 1592
Arbeiten mit Spark-Aufträgen ... 1609
Auftragsparameter ... 1610
Spark und PySpark Jobs ... 1620
Typen von Arbeitern ... 1778
Streaming-ETL-Aufträge ... 1783
Datensatzabgleich mit FindMatches ... 1798
Migrieren von Spark-Programmen ... 1838
Arbeiten mit Ray-Aufträgen ... 1846
Erste Schritte mit AWS Glue for Ray ... 1846
Unterstützte Ray-Laufzeitumgebungen ... 1847
Abrechnung für Worker in Ray-Aufträgen ... 1848
Parameter für Ray-Aufträge ... 1849
Auftragsmetriken von Ray ... 1852
Konfiguration der Python-Shell-Jobeigenschaften ... 1854
Einschränkungen ... 1854
Definieren von Auftragseigenschaften für Python-Shell-Aufträge ... 1855
Für Python-Shell-Aufträge unterstützte Bibliotheken ... 1857
Bereitstellen Ihrer eigenen Python-Bibliothek ... 1859
Verwendung AWS CloudFormation mit Python-Shell-Jobs in AWS Glue ... 1862
Migration von Python-Shell 3.6 auf Python-Shell 3.9 ... 1863
Migrieren Sie von AWS Glue Python-Shell-Jobs ... 1863

Überwachen 1865
AWS Schlagworte 1867
Automatisieren mit EventBridge 1872
Überwachung von -Ressourcen 1875
Protokollierung mit CloudTrail 1878
AWS Glue Streamen 1882
Anwendungsfälle für Streaming 1882
Was sind die Vorteile der Nutzung von AWS Glue Streaming? 1883
Wann sollte AWS Glue Streaming verwendet werden? 1884
Unterstützte Datenquellen 1885
Unterstützte Datenziele 1885
Tutorial: Erstellen Sie Ihren ersten Streaming-Workload mit AWS Glue Studio 1886
Voraussetzungen 1886
Streaming-Daten von Amazon Kinesis verarbeiten 1886
Tutorial: Erstellen Sie Ihren ersten Streaming-Workload mit AWS Glue Studio-Notebooks 1897
Voraussetzungen 1898
Streaming-Daten von Amazon Kinesis verarbeiten 1898
Streaming-Konzepte 1905
Aufbau eines AWS Glue Streaming-Jobs 1905
Streaming-Verbindungen 1909
Kafka-Verbindungen 1910
Kinesis-Verbindungen 1916
AWS Glue Automatische Streaming-Skalierung 1924
Aktivieren von Auto Scaling in AWS Glue Studio 1924
Auto Scaling mit der AWS CLI oder dem SDK aktivieren 1925
Funktionsweise 1926
Wartungsfenster 1928
Ein Wartungsfenster einrichten 1928
Verhalten des Wartungsfensters 1929
Überwachung von Job 1930
Umgang mit Datenverlust 1932
Advanced AWS Glue Streaming-Konzepte 1933
Zeitliche Überlegungen bei der Verarbeitung von Streams 1933
Windowing 1934
Umgang mit verspäteten Daten und Watermarks 1940
Überwachung von AWS Glue Streaming-Jobs 1942

AWS Glue Streaming-Metriken visualisieren ... 1943
AWS Glue Streaming-Metriken verwenden ... 1944
So erzielen Sie die beste Leistung ... 1949
Null-ETL-Integrationen ... 1951
Zero-ETL-Funktionen in AWS Glue ... 1951
Voraussetzungen ... 1952
Quellressourcen einrichten ... 1953
Zielressourcen einrichten ... 1954
Ein Amazon Redshift Data Warehouse erstellen ... 1959
Eine VPC einrichten ... 1960
Einrichtung einer kontenübergreifenden Integration ... 1964
Konfiguration einer Quelle ... 1965
Support für spezielle SAP-Entitäten ... 1965
Amazon DynamoDB DynamoDB-Quelle ... 1965
Salesforce-Quelle ... 1967
Salesforce MCAE-Quelle ... 1967
Zusätzliche Salesforce-Konfiguration ... 1968
OData SAP-Quelle ... 1968
ServiceNow Quelle ... 1969
Zendesk-Quelle ... 1969
Zoho CRM-Quelle ... 1969
Quelle für Facebook-Anzeigen ... 1971
Quelle für Instagram-Anzeigen ... 1971
Nicht unterstützte Salesforce-Felder ... 1972
Felder werden nicht unterstützt ServiceNow ... 1975
Konfigurieren eines Ziels ... 1980
SageMaker Lakehouse mit S3-Speicher ... 1981
Ziel für S3-Tabellen ... 1981
SageMaker Lakehouse mit Speicher Amazon Redshift ... 1985
Amazon Redshift Data Warehouse-Ziel ... 1986
Konfiguration des Integrationsziels ... 1988
Unverschachtelung von Partitionen und Schemas ... 1989
Standardverhalten zum Entfernen von Verschachtelungen und Partitionierungen ... 1990
Unverschachtelung von Schemas ... 1990
Datenpartitionierung ... 1994
Einschränkungen ... 2001

Allgemeine Aufgaben ... 2002
Eine Integration erstellen ... 2002
Eine Integration ändern ... 2004
Eine Integration löschen ... 2005
Verwenden APIs ... 2005
Überwachung einer Integration ... 2006
Integrationsstaaten ... 2006
Anzeigen von -Protokollen ... 2006
Anzeigen von -Metriken ... 2009
Ereignis-Benachrichtigungen ... 2009
Einschränkungen ... 2011
AWS Glue Qualität der Daten ... 2021
Vorteile und wichtige Features ... 2021
Funktionsweise ... 2022
Datenqualität für AWS Glue Data Catalog ... 2022
Datenqualität für AWS Glue ETL-Jobs ... 2023
Vergleich von AWS Glue Einstiegspunkten zur Datenqualität ... 2023
Überlegungen ... 2025
Terminologie ... 2025
Einschränkungen ... 2027
Versionshinweise für AWS Glue Datenqualität ... 2027
Allgemeine Verfügbarkeit: neue Features ... 2027
27. November 2023 (Vorschau) ... 2028
12. März 2024 ... 2028
26. Juni 2024 ... 2028
7. August 2024 ... 2028
22. November 2024 ... 2028
6. Dezember 2024 ... 2029
30. Juni 2025 ... 2029
Erkennung von Anomalien in AWS Glue Data Quality ... 2029
Funktionsweise ... 2030
Einzelheiten zum Algorithmus zur Erkennung von Anomalien ... 2035
IAM-Berechtigungen für AWS Glue Data Quality ... 2036
IAM-Berechtigungen ... 2037
Für die Planung von Auswertungsausführungen erforderliche IAM-Einrichtung ... 2040
IAM-Beispielrichtlinien ... 2041

Erste Schritte mit AWS Glue Data Quality für Data Catalog 2046
Voraussetzungen 2047
Step-by-step Beispiel 2047
Generieren von Regelempfehlungen 2047
Empfehlungen zu Überwachungsregeln 2049
Bearbeitung von empfohlenen Regelsätzen 2050
Erstellen eines neuen Regelsatzes 2051
Ausführen eines Regelsatzes zur Bewertung der Datenqualität 2053
Aufrufen des Datenqualitätswerts und der Ergebnisse 2054
Verwandte Themen 2055
Bewertung der Datenqualität mit AWS Glue Studio 2055
Vorteile 2056
Auswertung der Datenqualität für ETL-Aufträge in AWS Glue Studio 2056
Regelgenerator für Data Quality 2067
Konfiguration der Anomalieerkennung in AWS Glue ETL-Jobs 2072
Datenqualitätswerte und Anomalien anzeigen 2075
Datenqualität für ETL-Jobs in AWS Glue Studio-Notebooks 2079
Voraussetzungen 2080
Erstellen eines ETL-Auftrags in AWS Glue Studio 2080
Referenz zu Data Quality Definition Language (DQDL) 2085
Syntax 2087
Regeltypreferenz 2104
Verwendung APIs zur Messung und Verwaltung der Datenqualität 2165
Voraussetzungen 2165
Arbeiten mit AWS Glue-Datenqualitätsempfehlungen 2166
Mit AWS Glue Data Quality-Regelsätzen arbeiten 2169
Die Arbeit mit AWS Glue Data Quality läuft 2171
Arbeiten mit AWS Glue Data Quality-Ergebnissen 2176
Einrichten von Warnmeldungen, Bereitstellungen und Planung 2182
Benachrichtigungen und Benachrichtigungen in der EventBridge Amazon-Integration einrichten 2183
Richten Sie Benachrichtigungen und Benachrichtigungen in der CloudWatch Integration ein 2191
Abfragen von Datenqualitätsergebnissen 2193
Bereitstellung von Datenqualitätsregeln 2197
Planung von Datenqualitätsregeln 2197

Datenverschlüsselung im Ruhezustand für AWS Glue Data Quality ... 2197
AWS eigene Schlüssel ... 2197
Kundenverwaltete Schlüssel ... 2198
Erstellen Sie einen vom Kunden verwalteten Schlüssel ... 2201
Eine Sicherheitskonfiguration erstellen ... 2203
AWS Glue Data Quality-Verschlüsselungskontext ... 2204
Überwachung Ihrer Verschlüsselungsschlüssel für AWS Glue Data Quality ... 2205
Weitere Informationen ... 2210
Behebung von AWS Glue-Datenqualitätsfehlern ... 2210
Fehler: fehlendes Modul ... 2211
Fehler: unzureichende Berechtigungen ... 2211
Fehler: Regelsätze sind nicht eindeutig ... 2211
Fehler: Tabellen mit Sonderzeichen ... 2211
Fehler: Überlauf bei einem großen Regelsatz ... 2212
Fehler: der Regelstatus ist fehlgeschlagen ... 2212
AnalysisException: Die Existenz der Standarddatenbank konnte nicht überprüft werden ... 2212
Die bereitgestellte Schlüsselzuordnung ist für die angegebenen Datenrahmen nicht geeignet ... 2213
java.lang. RuntimeException : Daten konnten nicht abgerufen werden. ... 2213
STARTFEHLER: Fehler beim Herunterladen von S3 für den Bucket ... 2213
InvalidInputException (Status: 400): DataQuality Regeln können nicht analysiert werden .. 2214
Fehler: Eventbridge löst basierend auf dem von mir eingerichteten Zeitplan keine Glue-DQ-Aufträge aus ... 2214
CustomSQL-Fehler ... 2215
Dynamische Regeln ... 2215
Ausnahme in der Benutzerklasse: org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException ... 2218
UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing-Fehler: Es wurden keine Regeln oder Analysatoren bereitgestellt., keine praktikable Alternative bei der Eingabe ... 2218
Amazon Q-Datenintegration in AWS Glue ... 2219
Was ist Amazon Q? ... 2219
Amazon Q-Datenintegration in AWS Glue ... 2219
Verwenden der Amazon-Q-Datenintegration ... 2220
Bewährte Methoden ... 2222
Verbesserung des Service ... 2223
Überlegungen ... 2223

Einrichten der Amazon-Q-Datenintegration ... 2223
Konfigurieren von IAM-Berechtigungen ... 2223
Unterstützte Codegenerierung ... 2226
Beispiele für Interaktionen ... 2227
Amazon Q-Chat-Interaktionen ... 2227
AWS Glue Interaktionen mit Studio-Notizbüchern ... 2228
Kontextsensitivität nutzen ... 2232
Beispiel: Interaktionen ... 2232
Einschränkungen ... 2236
Orchestrierung ... 2237
Starten von Aufträgen und Crawlern über Auslöser ... 2237
AWS Glue triggers ... 2237
Hinzufügen von Auslösern ... 2240
Aktivieren und Deaktivieren von Auslösern ... 2244
Ausführen von komplexen ETL-Aktivitäten mithilfe von Blueprints und Workflows ... 2245
Übersicht über Workflows ... 2246
Manuelles Erstellen und Aufbauen eines Workflows ... 2250
Einen Workflow mit einem EventBridge Ereignis starten ... 2255
EventBridge Ereignisse anzeigen, die einen Workflow gestartet haben ... 2263
Ausführen und Überwachen eines Workflows ... 2264
Anhalten einer Workflow-Ausführung ... 2266
Reparieren und Fortsetzen einer Workflow-Ausführung ... 2267
Abrufen und Festlegen von Ausführungseigenschaften für Workflows ... 2274
Abfragen von Workflows mit dem AWS Glue API ... 2275
Blueprint- und Workflow-Einschränkungen ... 2280
Beheben von Blueprint-Fehlern ... 2281
Berechtigungen für Blueprint-Personas und -Rollen ... 2286
Entwickeln von Blueprints ... 2291
Übersicht über Blueprints ... 2291
Entwickeln von Blueprints ... 2295
Registrieren eines Blueprints ... 2320
Anzeigen von Blueprints ... 2323
Aktualisieren eines Blueprints ... 2325
Erstellen eines Workflows aus einem Blueprint ... 2327
Anzeigen von Blueprint-Ausführungen ... 2329
AWS CloudFormation für AWS Glue ... 2331

Beispieldatenbank ... 2333
Beispieldatenbank, -tabelle und partitionen ... 2334
Beispiel-Grok-Classifier ... 2338
Beispiel-JSON-Classifier ... 2339
Beispiel-XML-Classifier ... 2340
Beispiel für Amazon-S3-Crawler ... 2341
Beispielverbindung ... 2344
Beispiel für einen JDBC-Crawler ... 2345
Beispielauftrag für Amazon S3 nach Amazon S3 ... 2348
Beispielauftrag für JDBC an Amazon S3 ... 2349
Beispiel für einen Auslöser nach Bedarf ... 2351
Beispiel für einen geplanten Auslöser ... 2352
Beispiel für einen bedingten Auslöser ... 2353
Beispieltransformation für Machine Learning ... 2355
Beispiel für einen Regelsatz für die Datenqualität ... 2356
Beispiel für einen Regelsatz zur Datenqualität mit Scheduler EventBridge ... 2358
Beispiel für einen Entwicklungsendpunkt ... 2360
AWS Glue Leitfaden zur Programmierung ... 2362
Bereitstellen eigener, benutzerdefinierter Skripts ... 2362
AWS Glue für Spark ... 2363
Tutorial: Schreiben eines Spark-Skripts ... 2364
ETL rein PySpark ... 2378
ETL in Scala ... 2617
Features und Optimierungen ... 2704
AWS Glue für Ray ... 2981
Tutorial: Schreiben eines Ray-Skripts ... 2981
Verwenden von Ray Core und Ray Data in AWS Glue for Ray ... 2988
Bereitstellen von Dateien und Python-Bibliotheken ... 2990
Herstellen einer Verbindung zu Daten ... 2995
Arbeitet mit AWS SDKs ... 2998
AWS Glue API ... 3000
Sicherheit ... 3027
— Datentypen — ... 3027
DataCatalogEncryptionSettings ... 3028
EncryptionAtRest ... 3028
ConnectionPasswordEncryption ... 3029

EncryptionConfiguration ... 3030
S3Encryption ... 3030
CloudWatchEncryption ... 3031
JobBookmarksEncryption ... 3031
SecurityConfiguration ... 3031
GluePolicy ... 3032
DataQualityEncryption ... 3032
— Operationen — ... 3033
GetDataCatalogEncryptionSettings (get_data_catalog_encryption_settings) ... 3033
PutDataCatalogEncryptionSettings (put_data_catalog_encryption_settings) ... 3034
PutResourcePolicy (put_resource_policy) ... 3035
GetResourcePolicy (get_resource_policy) ... 3036
DeleteResourcePolicy (delete_resource_policy) ... 3037
CreateSecurityConfiguration (create_security_configuration) ... 3038
DeleteSecurityConfiguration (delete_security_configuration) ... 3039
GetSecurityConfiguration (get_security_configuration) ... 3039
GetSecurityConfigurations (get_security_configurations) ... 3040
GetResourcePolicies (get_resource_policies) ... 3041
Objekte katalogisieren ... 3042
Kataloge ... 3042
Datenbanken ... 3055
Tabellen ... 3065
Partitionen ... 3114
Verbindungen ... 3140
Benutzerdefinierte -Funktionen ... 3185
Einen Athena Katalog importieren ... 3192
Tabellenoptimierer ... 3194
— Datentypen — ... 3194
TableOptimizer ... 3195
TableOptimizerConfiguration ... 3196
TableOptimizerVpcConfiguration ... 3196
CompactionConfiguration ... 3197
IcebergCompactionConfiguration ... 3197
TableOptimizerRun ... 3198
BatchGetTableOptimizerEntry ... 3199
BatchTableOptimizer ... 3200

BatchGetTableOptimizerError ... 3201
RetentionConfiguration ... 3201
IcebergRetentionConfiguration ... 3201
OrphanFileDeletionConfiguration ... 3202
IcebergOrphanFileDeletionConfiguration ... 3202
CompactionMetrics ... 3203
RetentionMetrics ... 3203
OrphanFileDeletionMetrics ... 3203
IcebergCompactionMetrics ... 3203
IcebergRetentionMetrics ... 3204
IcebergOrphanFileDeletionMetrics ... 3204
RunMetrics ... 3205
— Operationen — ... 3205
GetTableOptimizer (get_table_optimizer) ... 3206
BatchGetTableOptimizer (batch_get_table_optimizer) ... 3207
ListTableOptimizerRuns (list_table_optimizer_runs) ... 3208
CreateTableOptimizer (create_table_optimizer) ... 3209
DeleteTableOptimizer (delete_table_optimizer) ... 3210
UpdateTableOptimizer (update_table_optimizer) ... 3211
Crawler und Classifier ... 3213
Classifier ... 3213
Crawler ... 3227
Spaltenstatistiken ... 3256
Scheduler ... 3273
Automatisches Generieren von ETL-Skripts ... 3275
— Datentypen — ... 3276
CodeGenNode ... 3276
CodeGenNodeArg ... 3276
CodeGenEdge ... 3277
Ort ... 3277
CatalogEntry ... 3278
MappingEntry ... 3278
— Operationen — ... 3279
CreateScript (create_script) ... 3279
GetDataflowGraph (get_dataflow_graph) ... 3280
GetMapping (get_mapping) ... 3280

GetPlan (get_plan) 3281
Visual Job API 3283
— Datentypen — 3283
CodeGenConfigurationNode 3287
JDBCConnectorOptionen 3293
StreamingDataPreviewOptions 3295
AthenaConnectorSource 3295
JDBCConnectorQuelle 3296
SparkConnectorSource 3297
CatalogSource 3298
Meine SQLCatalog Quelle 3298
SQLCatalogPostgre-Quelle 3299
SQLCatalogOracle-Quelle 3299
Microsoft SQLServer CatalogSource 3300
CatalogKinesisSource 3300
DirectKinesisSource 3301
KinesisStreamingSourceOptions 3302
CatalogKafkaSource 3304
DirectKafkaSource 3305
KafkaStreamingSourceOptions 3306
RedshiftSource 3308
AmazonRedshiftSource 3309
AmazonRedshiftNodeData 3309
AmazonRedshiftAdvancedOption 3312
Option 3312
S3 CatalogSource 3313
S3 SourceAdditionalOptions 3314
S3 CsvSource 3314
Direkt JDBCSource 3316
S3 DirectSourceAdditionalOptions 3317
S3 JsonSource 3318
S3 ParquetSource 3319
S3 DeltaSource 3321
S3 CatalogDeltaSource 3321
CatalogDeltaSource 3322
S3 HudiSource 3323

S3 CatalogHudiSource 3324
S3 ExcelSource 3324
CatalogHudiSource 3326
Dynamo-Quelle DBCatalog 3326
RelationalCatalogSource 3327
JDBCConnectorZiel 3327
SparkConnectorTarget 3328
BasicCatalogTarget 3329
Mein SQLCatalog Ziel 3330
SQLCatalogPostgre-Ziel 3330
Oracle SQLCatalog Target 3331
Microsoft SQLServer CatalogTarget 3331
RedshiftTarget 3332
AmazonRedshiftTarget 3333
UpsertRedshiftTargetOptions 3333
S3 CatalogTarget 3334
S3 GlueParquetTarget 3334
CatalogSchemaChangePolicy 3335
S3 DirectTarget 3336
S3 HudiCatalogTarget 3337
S3 HudiDirectTarget 3337
S3 DeltaCatalogTarget 3339
S3 DeltaDirectTarget 3339
S3 HyperDirectTarget 3341
S3 IcebergDirectTarget 3341
DirectSchemaChangePolicy 3342
ApplyMapping 3343
Mapping 3343
SelectFields 3345
DropFields 3345
RenameField 3345
Spigot 3346
Join 3347
JoinColumn 3347
SplitFields 3348
SelectFromCollection 3348

FillMissingValues 3349
Filter 3349
FilterExpression 3350
FilterValue 3350
CustomCode 3351
SparkSQL 3351
SqlAlias 3352
DropNullFields 3352
NullCheckBoxList 3353
NullValueField 3354
Datatype 3354
Mischen von 3354
Union 3355
PIIDetection 3355
Aggregate 3357
DropDuplicates 3357
GovernedCatalogTarget 3358
GovernedCatalogSource 3358
AggregateOperation 3359
GlueSchema 3359
GlueStudioSchemaColumn 3360
GlueStudioColumn 3360
DynamicTransform 3361
TransformConfigParameter 3362
EvaluateDataQuality 3363
DQResultsPublishingOptions 3363
DQStopJobOnFailureOptions 3364
EvaluateDataQualityMultiFrame 3364
Rezept 3365
RecipeReference 3366
SnowflakeNodeData 3366
SnowflakeSource 3369
SnowflakeTarget 3369
ConnectorDataSource 3369
ConnectorDataTarget 3370
RecipeStep 3371

RecipeAction ... 3372
ConditionExpression ... 3372
Aufträge ... 3373
Aufträge ... 3373
Auftragsausführungen ... 3399
Auslöser ... 3420
Integration APIs ... 3434
— Datentypen — ... 3434
Integration ... 3435
IntegrationConfig ... 3436
IntegrationPartition ... 3437
IntegrationError ... 3438
IntegrationFilter ... 3439
InboundIntegration ... 3439
SourceProcessingProperties ... 3440
TargetProcessingProperties ... 3440
SourceTableConfig ... 3441
TargetTableConfig ... 3441
— Operationen — ... 3442
CreateIntegration (Integration erstellen) ... 3442
ModifyIntegration (Integration modifizieren) ... 3445
DescribeIntegrations (beschreiben_Integrationen) ... 3448
DeleteIntegration (delete_integration) ... 3449
DescribeInboundIntegrations (describe_inbound_integrations) ... 3451
CreateIntegrationTableProperties (create_integration_table_properties) ... 3453
UpdateIntegrationTableProperties (update_integration_table_properties) ... 3454
GetIntegrationTableProperties (get_integration_table_properties) ... 3455
DeleteIntegrationTableProperties (delete_integration_table_properties) ... 3457
CreateIntegrationResourceProperty (create_integration_resource_property) ... 3457
UpdateIntegrationResourceProperty (update_integration_resource_property) ... 3459
GetIntegrationResourceProperty (get_integration_resource_property) ... 3460
UntagResource (untag_resource) ... 3461
ListTagsForResource (list_tags_for_resource) ... 3461
— Ausnahmen — ... 3462
ResourceNotFoundException ... 3462
InternalServerException ... 3463

IntegrationAlreadyExistsFault 3463
IntegrationConflictOperationFault 3463
IntegrationQuotaExceededFault 3463
KMSKeyNotAccessibleFault 3464
IntegrationNotFoundFault 3464
TargetResourceNotFound 3464
InvalidIntegrationStateFault 3464
Interaktive Sitzungen 3465
— Datentypen — 3465
Sitzung 3465
SessionCommand 3468
Statement 3468
StatementOutput 3469
StatementOutputData 3469
ConnectionsList 3470
— Operationen — 3470
CreateSession (Sitzung erstellen) 3470
StopSession (stop_session) 3474
DeleteSession (Sitzung löschen) 3475
GetSession (get_session) 3476
ListSessions (list_sessions) 3476
RunStatement (run_statement) 3478
CancelStatement (Aussage stornieren) 3479
GetStatement (get_statement) 3479
ListStatements (list_statements) 3480
DevEndpoints 3481
— Datentypen — 3481
DevEndpoint 3482
DevEndpointCustomLibraries 3486
— Operationen — 3487
CreateDevEndpoint (create_dev_endpoint) 3487
UpdateDevEndpoint (update_dev_endpoint) 3493
DeleteDevEndpoint (delete_dev_endpoint) 3494
GetDevEndpoint (get_dev_endpoint) 3495
GetDevEndpoints (get_dev_endpoints) 3496
BatchGetDevEndpoints (batch_get_dev_endpoints) 3497

ListDevEndpoints (list_dev_endpoints) 3498
Schema Registry 3499
— Datentypen — 3499
RegistryId 3499
RegistryListItem 3500
MetadataInfo 3501
OtherMetadataValueListItem 3501
SchemaListItem 3501
SchemaVersionListItem 3502
MetadataKeyValuePair 3503
SchemaVersionErrorItem 3503
ErrorDetails 3504
SchemaVersionNumber 3504
SchemaId 3504
— Operationen — 3505
CreateRegistry (create_registry) 3506
CreateSchema (Schema erstellen) 3507
GetSchema (get_schema) 3511
ListSchemaVersions (list_schema_versions) 3513
GetSchemaVersion (get_schema_version) 3514
GetSchemaVersionsDiff (get_schema_versions_diff) 3516
ListRegistries (list_registries) 3517
ListSchemas (list_schemas) 3518
RegisterSchemaVersion (register_schema_version) 3519
UpdateSchema (Schema aktualisieren) 3521
CheckSchemaVersionValidity (check_schema_version_validity) 3522
UpdateRegistry (update_registry) 3523
GetSchemaByDefinition (get_schema_by_definition) 3524
GetRegistry (get_registry) 3525
PutSchemaVersionMetadata (put_schema_version_metadata) 3527
QuerySchemaVersionMetadata (query_schema_version_metadata) 3528
RemoveSchemaVersionMetadata (remove_schema_version_metadata) 3530
DeleteRegistry (delete_registry) 3531
DeleteSchema (delete_schema) 3532
DeleteSchemaVersions (delete_schema_versions) 3533
Workflows 3535

— Datentypen — 3535
JobNodeDetails 3535
CrawlerNodeDetails 3536
TriggerNodeDetails 3536
Crawl 3536
Knoten 3537
Edge 3538
Workflow 3538
WorkflowGraph 3539
WorkflowRun 3540
WorkflowRunStatistics 3541
StartingEventBatchCondition 3542
Blueprint 3542
BlueprintDetails 3544
LastActiveDefinition 3544
BlueprintRun 3545
— Operationen — 3546
CreateWorkflow (create_workflow) 3547
UpdateWorkflow (update_workflow) 3549
DeleteWorkflow (delete_workflow) 3550
GetWorkflow (get_workflow) 3551
ListWorkflows (list_workflows) 3551
BatchGetWorkflows (batch_get_workflows) 3552
GetWorkflowRun (get_workflow_run) 3553
GetWorkflowRuns (get_workflow_runs) 3554
GetWorkflowRunProperties (get_workflow_run_properties) 3555
PutWorkflowRunProperties (put_workflow_run_properties) 3556
CreateBlueprint (create_blueprint) 3557
UpdateBlueprint (update_blueprint) 3558
DeleteBlueprint (delete_blueprint) 3559
ListBlueprints (list_blueprints) 3560
BatchGetBlueprints (batch_get_blueprints) 3561
StartBlueprintRun (start_blueprint_run) 3562
GetBlueprintRun (get_blueprint_run) 3562
GetBlueprintRuns (get_blueprint_runs) 3563
StartWorkflowRun (start_workflow_run) 3564

StopWorkflowRun (stop_workflow_run) 3565
ResumeWorkflowRun (resume_workflow_run) 3566
Nutzungsprofile 3567
— Datentypen — 3567
ProfileConfiguration 3567
ConfigurationObject 3568
UsageProfileDefinition 3569
— Operationen — 3569
CreateUsageProfile (create_usage_profile) 3569
GetUsageProfile (get_usage_profile) 3571
UpdateUsageProfile (update_usage_profile) 3572
DeleteUsageProfile (delete_usage_profile) 3572
ListUsageProfiles (list_usage_profiles) 3573
Machine Learning 3574
— Datentypen — 3574
TransformParameters 3575
EvaluationMetrics 3575
MLTransform 3576
FindMatchesParameters 3579
FindMatchesMetrics 3580
ConfusionMatrix 3582
GlueTable 3582
TaskRun 3583
TransformFilterCriteria 3585
TransformSortCriteria 3586
TaskRunFilterCriteria 3586
TaskRunSortCriteria 3587
TaskRunProperties 3587
FindMatchesTaskRunProperties 3588
ImportLabelsTaskRunProperties 3588
ExportLabelsTaskRunProperties 3589
LabelingSetGenerationTaskRunProperties 3589
SchemaColumn 3589
TransformEncryption 3590
MLUserDataEncryption 3590
ColumnImportance 3591

— Operationen — 3591
Erstellen (create_ml_transform) MLTransform 3592
Aktualisieren (update_ml_transform) MLTransform 3596
MLTransform Löschen (delete_ml_transform) 3598
Holen Sie sich (get_ml_transform) MLTransform 3599
Holen MLTransforms Sie sich (get_ml_transforms) 3602
MLTransforms Liste (list_ml_transforms) 3603
Starten MLEvaluation TaskRun (start_ml_evaluation_task_run) 3604
Start (MLLabelingSetGenerationTaskRun start_ml_labeling_set_generation_task_run) 3605
Holen Sie sich MLTask Run (get_ml_task_run) 3607
MLTaskRuns abrufen (get_ml_task_runs) 3608
MLTaskAusführung abbrechen (cancel_ml_task_run) 3610
StartExportLabelsTaskRun (start_export_labels_task_run) 3611
StartImportLabelsTaskRun (start_import_labels_task_run) 3612
Data Quality 3613
— Datentypen — 3613
DataSource 3614
DataQualityRulesetListDetails 3615
DataQualityTargetTable 3615
DataQualityRulesetEvaluationRunDescription 3616
DataQualityRulesetEvaluationRunFilter 3617
DataQualityEvaluationRunAdditionalRunOptions 3617
DataQualityRuleRecommendationRunDescription 3617
DataQualityRuleRecommendationRunFilter 3618
DataQualityResult 3618
DataQualityAnalyzerResult 3620
DataQualityObservation 3621
MetricBasedObservation 3621
DataQualityMetricValues 3622
DataQualityRuleResult 3622
DataQualityResultDescription 3624
DataQualityResultFilterCriteria 3624
DataQualityRulesetFilterCriteria 3625
DataQualityAggregatedMetrics 3626
StatisticAnnotation 3626
TimestampedInclusionAnnotation 3627

AnnotationError 3627
DatapointInclusionAnnotation 3628
StatisticSummaryList 3628
StatisticSummary 3628
RunIdentifier 3630
StatisticModelResult 3630
— Operationen — 3631
StartDataQualityRulesetEvaluationRun (start_data_quality_ruleset_evaluation_run) 3632
CancelDataQualityRulesetEvaluationRun (cancel_data_quality_ruleset_evaluation_run) 3634
GetDataQualityRulesetEvaluationRun (get_data_quality_ruleset_evaluation_run) 3634
ListDataQualityRulesetEvaluationRuns (list_data_quality_ruleset_evaluation_runs) 3636
StartDataQualityRuleRecommendationRun (start_data_quality_rule_recommendation_run) 3637
CancelDataQualityRuleRecommendationRun
(cancel_data_quality_rule_recommendation_run) 3639
GetDataQualityRuleRecommendationRun (get_data_quality_rule_recommendation_run) .. 3640
ListDataQualityRuleRecommendationRuns (list_data_quality_rule_recommendation_runs) 3642
GetDataQualityResult (get_data_quality_result) 3642
BatchGetDataQualityResult (batch_get_data_quality_result) 3645
ListDataQualityResults (list_data_quality_results) 3645
CreateDataQualityRuleset (create_data_quality_ruleset) 3646
DeleteDataQualityRuleset (delete_data_quality_ruleset) 3648
GetDataQualityRuleset (get_data_quality_ruleset) 3649
ListDataQualityRulesets (list_data_quality_rulesets) 3650
UpdateDataQualityRuleset (update_data_quality_ruleset) 3651
ListDataQualityStatistics (list_data_quality_statistics) 3652
TimestampFilter 3653
CreateDataQualityRulesetRequest 3654
GetDataQualityRulesetResponse 3655
GetDataQualityResultResponse 3656
StartDataQualityRuleRecommendationRunRequest 3658
GetDataQualityRuleRecommendationRunResponse 3659
BatchPutDataQualityStatisticAnnotation (batch_put_data_quality_statistic_annotation) 3660
GetDataQualityModel (get_data_quality_model) 3661
GetDataQualityModelResult (get_data_quality_model_result) 3662
ListDataQualityStatisticAnnotations (list_data_quality_statistic_annotations) 3663
PutDataQualityProfileAnnotation (put_data_quality_profile_annotation) 3664

Sensible Daten 3665
— Datentypen — 3665
CustomEntityType 3665
— Operationen — 3666
CreateCustomEntityType (create_custom_entity_type) 3666
DeleteCustomEntityType (delete_custom_entity_type) 3667
GetCustomEntityType (get_custom_entity_type) 3668
BatchGetCustomEntityTypes (batch_get_custom_entity_types) 3669
ListCustomEntityTypes (list_custom_entity_types) 3670
Markieren APIs 3671
— Datentypen — 3671
Markierung 3671
— Operationen — 3671
TagResource (tag_resource) 3672
UntagResource (untag_resource) 3672
GetTags (get_tags) 3673
Gängige Datentypen 3674
Markierung 3674
DecimalNumber 3675
ErrorDetail 3675
PropertyPredicate 3675
ResourceUri 3676
ColumnStatistics 3676
ColumnStatisticsError 3677
ColumnError 3677
ColumnStatisticsData 3677
BooleanColumnStatisticsData 3678
DateColumnStatisticsData 3679
DecimalColumnStatisticsData 3679
DoubleColumnStatisticsData 3680
LongColumnStatisticsData 3680
StringColumnStatisticsData 3681
BinaryColumnStatisticsData 3681
Zeichenfolgemuster 3681
Ausnahmen 3684
AccessDeniedException 3684

AlreadyExistsException 3684
ConcurrentModificationException 3685
ConcurrentRunsExceededException 3685
CrawlerNotRunningException 3685
CrawlerRunningException 3685
CrawlerStoppingException 3686
EntityNotFoundException 3686
FederationSourceException 3686
FederationSourceRetryableException 3687
GlueEncryptionException 3687
IdempotentParameterMismatchException 3687
IllegalWorkflowStateException 3687
InternalServiceException 3688
InvalidExecutionEngineException 3688
InvalidInputException 3688
InvalidStateException 3689
InvalidTaskStatusTransitionException 3689
JobDefinitionErrorException 3689
JobRunInTerminalStateException 3689
JobRunInvalidStateTransitionException 3690
JobRunNotInTerminalStateException 3690
LateRunnerException 3690
NoScheduleException 3691
OperationTimeoutException 3691
ResourceNotReadyException 3691
ResourceNumberLimitExceededException 3691
SchedulerNotRunningException 3692
SchedulerRunningException 3692
SchedulerTransitioningException 3692
UnrecognizedRunnerException 3692
ValidationException 3693
VersionMismatchException 3693
AWS Glue API-Codebeispiele 3694
Grundlagen 3704
Hallo AWS Glue 3705
Erlernen der Grundlagen 3715

Aktionen ... 3853
Sicherheit ... 3997
Datenschutz ... 3998
Verschlüsseln von Daten im Ruhezustand ... 3998
Verschlüsseln von Daten während der Übertragung ... 4018
Compliance mit FIPS ... 4018
Schlüsselverwaltung ... 4018
AWS Glue Abhängigkeit von anderen AWS Diensten ... 4019
Entwicklungsendpunkte ... 4019
Identity and Access Management ... 4020
Zielgruppe ... 4021
Authentifizierung mit Identitäten ... 4022
Verwalten des Zugriffs mit Richtlinien ... 4026
So funktioniert AWS Glue mit IAM ... 4029
Konfiguration von IAM-Berechtigungen für AWS Glue ... 4038
AWS Beispiele für Richtlinien zur Glue-Zugriffskontrolle ... 4073
Gewährung AWS verwalteter Richtlinien ... 4101
Gewährung von Richtlinien mit dynamischem Geltungsbereich ... 4111
AWS GlueRessource angeben ARNs ... 4112
Gewährung von kontenübergreifendem Zugriff ... 4122
Fehlerbehebung ... 4130
AWS Lake Formation Modelle für die Zugriffskontrolle ... 4132
Verwenden von AWS Glue with AWS Lake Formation für den vollständigen Tabellenzugriff ... 4132
Lake Formation für FGAC ... 4145
Verwenden von Amazon S3 Access Grants mit AWS Glue ... 4158
Wie AWS Glue funktioniert mit S3 Access Grants ... 4158
Überlegungen zu S3 Access Grants mit AWS Glue ... 4159
Richten Sie S3 Access Grants ein mit AWS Glue ... 4160
Protokollierung und Überwachung ... 4162
Compliance-Validierung ... 4163
Ausfallsicherheit ... 4164
Sicherheit der Infrastruktur ... 4165
Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS PrivateLink für AWS Glue ... 4165
Konfiguration von Shared Amazon VPCs ... 4168
Fehlerbehebung AWS Glue ... 4169

Informationen AWS Glue zur Problembehandlung sammeln 4169
Fehlerbehebung bei Spark-Fehlern 4170
Fehler: Ressource nicht verfügbar 4171
Fehler: S3-Endpunkt oder NAT-Gateway für SubnetId wurde in VPC nicht gefunden 4171
Fehler: Eingehende Regel in der Sicherheitsgruppe erforderlich 4172
Fehler: Ausgehende Regel in der Sicherheitsgruppe erforderlich 4172
Fehler: Die Auftragsausführung ist fehlgeschlagen, da der übergebenen Rolle die Rollenübernahme für den AWS Glue Dienst erteilt werden sollte 4172
Fehler: Die DescribeVpcEndpoints Aktion ist nicht autorisiert. die VPC-ID kann nicht validiert werden vpc-id 4173
Fehler: Die DescribeRouteTables Aktion ist nicht autorisiert. Die Subnetz-ID konnte nicht überprüft werden: Subnetz-ID in VPC-ID: vpc-id 4173
Fehler: ec2 konnte nicht aufgerufen werden: DescribeSubnets 4173
Fehler: ec2 konnte nicht aufgerufen werden: DescribeSecurityGroups 4173
Fehler: Subnetz für AZ wurde nicht gefunden 4173
Fehler: Ausnahme der Auftragsausführung beim Schreiben in ein JDBC-Ziel 4174
Fehler: Amazon S3: Der Vorgang ist für die Speicherklasse des Objekts nicht gültig 4174
Fehler: Amazon S3 Zeitüberschreitung 4175
Fehler: Amazon-S3-Zugriff verweigert 4175
Fehler: Amazon-S3-Zugriffsschlüssel-ID ist nicht vorhanden 4175
Fehler: Auftragsausführung schlägt fehl, wenn auf Amazon S3 mit einem `s3a://`-URI zugegriffen wird 4176
Fehler: Amazon-S3-Servicetoken abgelaufen 4178
Fehler: Keine private DNS für die Netzwerkschnittstelle gefunden 4178
Fehler: Bereitstellung des Entwicklungsendpunkts fehlgeschlagen 4178
Fehler: Notebook-Server-ERSTELLUNG_FEHLGESCHLAGEN 4178
Fehler: Lokales Notebook kann nicht gestartet werden 4179
Fehler: Ausführen von Crawler fehlgeschlagen 4179
Fehler: Partitionen wurden nicht aktualisiert 4179
Fehler: Aktualisierung des Auftragslesezeichens aufgrund einer Versionsabweichung fehlgeschlagen 4180
Fehler: Ein Auftrag verarbeitet Daten erneut, wenn Auftragslesezeichen aktiviert sind 4181
Fehler: Failover-Verhalten zwischen VPCs AWS Glue 4182
Crawler-Fehler, wenn der Crawler Lake Formation Formation-Berechtigungen verwendet 4183
Fehler: Der S3-Standort: s3://examplepath ist nicht registriert 4183

Fehler: User/Role ist nicht autorisiert, Folgendes auszuführen: lakeformation: on resource GetDataAccess ... 4183
Fehler: Unzureichende Lake-Formation-Berechtigung(en) für (Datenbankname: exampleDatabase, Tabellenname: exampleTable) ... 4184
Fehler: Unzureichende Lake-Formation-Berechtigung(en) auf s3://examplepath ... 4184
Häufig gestellte Fragen zur Crawler-Konfiguration mit Lake-Formation-Anmeldeinformationen ... 4185
Fehlerbehebung bei Ray-Fehlern ... 4186
Überprüfung von Ray-Auftragsprotokollen ... 4186
Fehlerbehebung bei Ray-Auftragsfehlern ... 4187
AWS Glue Ausnahmen für maschinelles Lernen ... 4189
Stornieren MLTask RunActivity ... 4189
Erstellen MLTask RunActivity ... 4189
MLTransformAktivität löschen ... 4190
Holen MLTask RunActivity ... 4191
Holen MLTask RunsActivity ... 4191
MLTransformAktivität abrufen ... 4191
MLTransformsAktivität abrufen ... 4192
GetSaveLocationForTransformArtifactActivity ... 4192
GetTaskRunArtifactActivity ... 4193
Veröffentlichen MLTransform ModelActivity ... 4193
PullLatestMLTransformModelActivity ... 4194
PutJobMetadataForMLTransformAktivität ... 4194
StartExportLabelsTaskRunActivity ... 4195
StartImportLabelsTaskRunActivity ... 4195
Fangen MLEvaluation TaskRunActivity ... 4196
Fangen MLLabeling SetGenerationTaskRunActivity ... 4197
MLTransformAktivität aktualisieren ... 4198
AWS Glue Kontingente ... 4199
Verbesserung der AWS Glue Leistung ... 4200
Optimierte Strategien für Ihren Jobtyp ... 4200
Verbessern der Spark-Leistung ... 4200
Optimieren von Lesevorgängen mit Pushdown ... 4201
Prädikat-Pushdown für auf Amazon S3 gespeicherte Dateien ... 4201
Pushdown beim Arbeiten mit JDBC-Quellen ... 4202
Hinweise und Einschränkungen für Pushdown in AWS Glue ... 4205

Verwenden von Auto Scaling für AWS Glue 4206
Voraussetzungen 4207
Aktivieren von Auto Scaling in AWS Glue Studio 1924
Auto Scaling mit der AWS CLI oder dem SDK aktivieren 1925
Auto Scaling mit interaktiven Sitzungen aktivieren 4209
Tipps und Überlegungen 4209
Überwachung von Auto Scaling mit CloudWatch Amazon-Metriken 4210
Überwachung von Auto Scaling mit Amazon CloudWatch Logs 4211
Überwachen von Auto Scaling mit der Spark-Benutzeroberfläche 4212
Überwachen der DPU-Nutzung bei der Auto-Scaling-Auftragsausführung 4213
Einschränkungen 4213
Workload-Partitionierung mit begrenzter Ausführung 4213
Aktivieren der Workload-Partitionierung 4213
Einrichtung einer AWS Glue Auslöser, um den Job automatisch auszuführen 4215
Bekannte Probleme 4216
Verhindern des auftragsübergreifenden Datenzugriffs 4216
Dokumentationsverlauf 4219
Frühere Aktualisierungen 4289
AWS Glossar 4291
..... mmmmccxcii

Was ist AWS Glue?

AWS Glue ist ein serverloser Datenintegrationsdienst, der es Analytics-Benutzern erleichtert, Daten aus mehreren Quellen zu ermitteln, aufzubereiten, zu verschieben und zu integrieren. Sie können es für Analysen, Machine Learning und Anwendungsentwicklung verwenden. Es umfasst auch zusätzliche Produktivitäts- und Datenops-Tools für die Erstellung, Ausführung von Aufträgen und die Implementierung von Geschäftsabläufen.

Mit AWS Glue, können Sie mehr als 70 verschiedene Datenquellen entdecken und eine Verbindung zu ihnen herstellen und Ihre Daten in einem zentralen Datenkatalog verwalten. Sie können ETL-Pipelines (Extract, Transform, Load) visuell erstellen, ausführen und überwachen, um Daten in Ihre Date Lakes zu laden. Außerdem können Sie mithilfe von Amazon Athena, Amazon EMR und Amazon Redshift Spectrum sofort katalogisierte Daten durchsuchen und abfragen.

AWS Glue konsolidiert wichtige Datenintegrationsfunktionen in einem einzigen Service.
Dazu gehören Data Discovery, moderne ETL, Bereinigung, Transformation und zentralisierte Katalogisierung. Es ist außerdem Serverless, was bedeutet, dass keine Infrastruktur verwaltet werden muss. Mit flexibler Unterstützung für alle Workloads wie ETL, ELT und Streaming in einem Service AWS Glue unterstützt Benutzer bei verschiedenen Workloads und Benutzertypen.

Außerdem AWS Glue macht es einfach, Daten in Ihre gesamte Architektur zu integrieren. Es lässt sich in AWS Analysedienste und Amazon S3 S3-Datenseen integrieren. AWS Glue verfügt über Integrationsschnittstellen und Tools zur Erstellung von Jobs, die für alle Benutzer, von Entwicklern bis hin zu Geschäftsanwendern, einfach zu bedienen sind und maßgeschneiderte Lösungen für unterschiedliche technische Fähigkeiten bieten.

Mit der Fähigkeit, bei Bedarf zu skalieren, AWS Glue hilft Ihnen dabei, sich auf hochwertige Aktivitäten zu konzentrieren, die den Wert Ihrer Daten maximieren. Es skaliert für jede Datengröße und unterstützt alle Datentypen und Schemavarianzen. Um die Agilität zu erhöhen und die Kosten zu optimieren, AWS Glue bietet integrierte Hochverfügbarkeit und pay-as-you-go Abrechnung.

Preisinformationen finden Sie unter AWS Glue Preisgestaltung.

AWS Glue Studio

AWS Glue Studio ist eine grafische Oberfläche, die es einfach macht, Datenintegrationsjobs zu erstellen, auszuführen und zu überwachen AWS GlueSie können Workflows für die Datentransformation visuell erstellen und nahtlos auf der Apache-Spark-basierten Serverless-ETL-Engine in laufen lassen. AWS Glue.

Mit AWS Glue Studio, können Sie Jobs erstellen und verwalten, die Daten sammeln, transformieren und bereinigen. Sie können auch verwenden AWS Glue Studio zur Fehlerbehebung und Bearbeitung von Jobskripten.

Themen

- AWS Glue features
- Erfahren Sie mehr über Innovationen in AWS Glue
- Erste Schritte mit AWS Glue
- Zugriff AWS Glue
- Zugehörige Services

AWS Glue features

AWS Glue Funktionen lassen sich in drei Hauptkategorien einteilen:

- Entdecken und organisieren von Daten
- Transformieren, vorbereiten und bereinigen von Daten für die Analyse
- Erstellen und Überwachen von Datenpipelines

Entdecken und organisieren von Daten

- Vereinheitlichen und durchsuchen Sie mehrere Datenspeicher — Speichern, indexieren und durchsuchen Sie mehrere Datenquellen und Datenspeicher, indem Sie alle Ihre Daten katalogisieren. AWS
- Automatisches Auffinden von Daten — Verwenden AWS Glue Crawler, um automatisch Schemainformationen abzuleiten und sie in Ihre zu integrieren. AWS Glue Data Catalog
- Schemas und Berechtigungen verwalten – Validieren und kontrollieren Sie den Zugriff auf Ihre Datenbanken und Tabellen.
- Stellen Sie eine Connect zu einer Vielzahl von Datenquellen her — Nutzen Sie mehrere Datenquellen, sowohl vor Ort als auch vor Ort AWS, mit AWS Glue Verbindungen zum Aufbau Ihres Data Lakes.

Transformieren, vorbereiten und bereinigen von Daten für Analysen

- Visuelle Transformation von Daten mit einer Job-Canvas-Oberfläche — Definieren Sie Ihren ETL-Prozess im Visual Job Editor und generieren Sie automatisch den Code zum Extrahieren, Transformieren und Laden Ihrer Daten.
- Erstellen Sie komplexe ETL-Pipelines mit einfacher Jobplanung — Invoke AWS Glue Jobs nach einem Zeitplan, auf Abruf oder auf der Grundlage eines Ereignisses.
- Reinigen und transformieren Sie Streaming-Daten während der Übertragung – Ermöglichen Sie kontinuierlichen Datenverbrauch und bereinigen und transformieren Sie ihn während der Übertragung. Dadurch steht es in Sekundenschnelle für Analysen in Ihrem Zieldatenspeicher zur Verfügung.
- Daten deduplizieren und bereinigen mit integriertem Machine Learning – Bereinigen und bereiten Sie Ihre Daten mit dem `FindMatches`-Feature für die Analyse vor, ohne Experte für Machine Learning zu werden. Dieses Feature dedupliziert und findet Datensätze, die nicht perfekt zueinander passen.
- Integrierte Job-Notizbücher — AWS Glue Job-Notebooks bieten serverlose Notebooks mit minimalem Einrichtungsaufwand AWS Glue damit Sie schnell loslegen können.
- ETL-Code bearbeiten, debuggen und testen — Mit AWS Glue In interaktiven Sitzungen können Sie Daten interaktiv untersuchen und aufbereiten. Mit der IDE oder dem Notebook Ihrer Wahl können Sie Daten interaktiv erkunden, ausprobieren und verarbeiten.
- Sensible Daten definieren, erkennen und korrigieren — AWS Glue Mit der Erkennung sensibler Daten können Sie sensible Daten in Ihrer Datenpipeline und in Ihrem Data Lake definieren, identifizieren und verarbeiten.

Erstellen und Überwachen von Datenpipelines

- Automatische Skalierung basierend auf Workload – Skalieren Sie Ressourcen basierend auf der Arbeitslast dynamisch nach oben und unten. Dadurch werden Arbeitern nur bei Bedarf Jobs zugewiesen.
- Automatisieren Sie Jobs mit ereignisbasierten Triggern — Starten Sie Crawler oder AWS Glue Jobs mit ereignisbasierten Triggern und Entwerfen Sie eine Kette von abhängigen Jobs und Crawlern.
- Jobs ausführen und überwachen — Ausführen AWS Glue Jobs mit Engine Ihrer Wahl, Spark oder Ray. Überwachen Sie sie mit automatisierten Überwachungstools, AWS Glue Einblicke in die Auftragsausführung und AWS CloudTrail. Verbessern Sie Ihre Überwachung von Spark-gestützten Aufträgen mit der Apache-Spark-Benutzeroberfläche.

- Definieren Sie Workflows für ETL- und Integrationsaktivitäten – Definieren Sie Workflows für ETL und Integrationsaktivitäten für mehrere Crawler, Jobs und Auslöser.

Erfahren Sie mehr über Innovationen in AWS Glue

Erfahren Sie mehr über die neuesten Innovationen AWS Glue und erfahren Sie, wie Kunden AWS Glue die Self-Service-Datenaufbereitung in ihrem gesamten Unternehmen nutzen.

Erfahren Sie, wie Kunden AWS Glue über das herkömmliche Setup hinaus skalieren und wie sie AWS Glue die Auftragsüberwachung und Leistung konfigurieren.

Erste Schritte mit AWS Glue

Wir empfehlen Ihnen, dass Sie mit den folgenden Abschnitten beginnen:

- Überblick über die Verwendung AWS Glue
- AWS Glue Konzepte
- Einrichtung von IAM-Berechtigungen für AWS Glue
- Erste Schritte mit dem AWS Glue Data Catalog
- Autorenjobs in AWS Glue
- Erste Schritte mit AWS Glue interaktive Sitzungen
- Orchestrierung in AWS Glue

Zugriff AWS Glue

Sie können Ihre erstellen, anzeigen und verwalten AWS Glue Jobs mit den folgenden Schnittstellen:

- AWS Glue Konsole — Bietet eine Weboberfläche, über die Sie Ihre erstellen, anzeigen und verwalten können AWS Glue Jobs. Informationen zum Zugriff auf die Konsole finden Sie unter AWS Glue.
- AWS Glue Studio— Bietet eine grafische Oberfläche, über die Sie Ihre erstellen und bearbeiten können AWS Glue Jobs visuell. Weitere Informationen finden Sie unter Erstellung visueller ETL-Jobs.
- AWS Glue Abschnitt der AWS CLI Referenz — Enthält AWS CLI Befehle, die Sie verwenden können mit AWS Glue. Weitere Informationen finden Sie unter AWS CLI Referenz für AWS Glue.

- AWS Glue API — Bietet eine vollständige API-Referenz für Entwickler. Weitere Informationen finden Sie unter AWS Glue API.

Zugehörige Services

Nutzer von AWS Glue verwenden auch:

- AWS Lake Formation— Ein Dienst, bei dem es sich um eine Autorisierungsebene handelt, die eine differenzierte Zugriffskontrolle auf Ressourcen in der AWS Glue Data Catalog.
- AWS Glue DataBrew— Ein visuelles Datenvorbereitungstool, mit dem Sie Daten bereinigen und normalisieren können, ohne Code schreiben zu müssen.

AWS Glue: Funktionsweise

AWS Glue verwendet andere AWS Dienste, um Ihre ETL-Jobs (Extrahieren, Transformieren und Laden) zu orchestrieren, um Data Warehouses und Data Lakes aufzubauen und Output-Streams zu generieren. AWS Glue ruft API-Operationen auf, um Ihre Daten zu transformieren, Laufzeitprotokolle zu erstellen, Ihre Joblogik zu speichern und Benachrichtigungen zu erstellen, mit denen Sie Ihre Jobausführungen überwachen können. Das Tool AWS Glue Die Konsole verbindet diese Dienste zu einer verwalteten Anwendung, sodass Sie sich auf die Erstellung und Überwachung Ihrer ETL-Arbeit konzentrieren können. Die Konsole führt in Ihrem Namen Verwaltungs- und Auftragsentwicklungsoperationen durch. Sie geben Anmeldeinformationen und andere Eigenschaften an AWS Glue um auf Ihre Datenquellen zuzugreifen und in Ihre Datenziele zu schreiben.

AWS Glue kümmert sich um die Bereitstellung und Verwaltung der Ressourcen, die für die Ausführung Ihres Workloads erforderlich sind. Sie müssen die Infrastruktur für ein ETL-Tool nicht erstellen, weil AWS Glue macht es für Sie. Wenn Ressourcen benötigt werden, um die Startzeit zu reduzieren, AWS Glue verwendet eine Instanz aus seinem warmen Instanzenpool, um Ihren Workload auszuführen.

Mit AWS Glue, Sie erstellen Jobs mithilfe von Tabellendefinitionen in Ihrem Datenkatalog. Jobs bestehen aus Skripten, die die Anweisungen enthalten, mit denen die gewünschten Datentransformationsaufgaben ausgeführt werden. Sie verwenden Auslöser, um Aufträge entweder nach einem Zeitplan oder als Ergebnis eines bestimmten Ereignisses zu initiieren. Sie legen fest, wo sich Ihre Zieldaten befinden und welche Quelldaten Ihr Ziel befüllen. Basierend auf Ihren Eingaben AWS Glue transformiert Ihre Daten vom Quell- in das Zielformat. Alternativ können Sie auch benutzerdefinierte Skripts in der AWS Glue Konsole oder API, um Ihre Daten gemäß Ihren spezifischen Anforderungen zu verarbeiten.

Datenquellen und -ziele

AWS Glue for Spark ermöglicht es Ihnen, Daten aus mehreren Systemen und Datenbanken zu lesen und zu schreiben, darunter:

- Amazon S3
- Amazon-DynamoDB
- Amazon Redshift
- Amazon Relational Database Service (Amazon RDS)
- Über JDBC zugängliche Datenbanken von Drittanbietern

- MongoDB und Amazon DocumentDB (mit MongoDB-Kompatibilität)
- Andere Marketplace-Konnektoren und Apache-Spark-Plugins

Datenströme

AWS Glue for Spark kann Daten aus den folgenden Systemen streamen:

- Amazon Kinesis Data Streams
- Apache Kafka

AWS Glue ist in mehreren AWS Regionen verfügbar. Weitere Informationen finden Sie unter AWS Regionen und Endpunkte in der Allgemeine Amazon Web Services-Referenz.

Themen

- Isolierte, Serverless-ETL-Aufträge
- AWS Glue Konzepte
- AWS Glue Komponenten
- AWS Glue für Spark und AWS Glue für Ray
- Umwandlung von halbstrukturierten Schemas in relationale Schemas mit AWS Glue
- AWS Systeme vom Typ Glue

Isolierte, Serverless-ETL-Aufträge

AWS Glue führt Ihre ETL-Jobs in einer serverlosen Umgebung mit Engine, Spark oder Ray Ihrer Wahl aus. AWS Glue führt diese Jobs auf virtuellen Ressourcen aus, die es in seinem eigenen Dienstkonto bereitstellt und verwaltet.

AWS Glue ist für Folgendes konzipiert:

- Separieren von Kundendaten.
- Schützen von Kundendaten während der Übertragung und im Speicherzustand.
- Zugriff nur dann auf Kundendaten, wenn dies aufgrund von Kundenanfragen erforderlich ist (über temporäre, auf einen bestimmten Umfang beschränkte Berechtigungsnachweise oder mit Zustimmung eines Kunden zu IAM-Rollen in seinem Konto).

Bei der Bereitstellung eines ETL-Auftrags stellen Sie Eingangsdatenquellen und Ausgangsdatenziele in Ihrer Virtual Private Cloud (VPC) zur Verfügung. Zusätzlich stellen Sie die IAM-Rolle, die VPC-ID, die Subnetz-ID und die Sicherheitsgruppe zur Verfügung, die für den Zugriff auf Datenquellen und -ziele benötigt werden. Für jedes Tupel (Kundenkonto-ID, IAM-Rolle, Subnetz-ID und Sicherheitsgruppe) AWS Glue erstellt eine neue Umgebung, die auf Netzwerk- und Managementebene von allen anderen Umgebungen in Ihrem AWS Glue Dienstkonto.

Sie erstellen und konfigurieren AWS Glue Ressourcen wie Datenkataloge, Jobs und Crawler in Ihrem AWS Konto. Diese Ressourcen werden dann der IAM-Rolle und den Netzwerkeinstellungen (Subnetz und Sicherheitsgruppe) zugeordnet, die Sie während des Erstellungsprozesses angeben.

AWS Glue erstellt elastische Netzwerkschnittstellen in Ihrem Subnetz mithilfe privater IP-Adressen. Aufträge nutzen diese Elastic-Network-Schnittstellen, um auf Ihre Datenquellen und Datenziele zuzugreifen. Der eingehende, ausgehende und innerhalb der Job-Run-Umgebung wird durch Ihre VPC- und Netzwerkrichtlinien geregelt, mit einer Ausnahme: Aufrufe an AWS Glue Bibliotheken können den Datenverkehr weiterleiten an AWS Glue API-Operationen über die AWS Glue VPC. Alle AWS Glue API-Aufrufe werden protokolliert. Somit können Dateneigentümer den API-Zugriff überprüfen, indem sie die Aktivierung aktivieren AWS CloudTrail, wodurch Auditprotokolle an Ihr Konto gesendet werden.

AWS Glue verwaltete Umgebungen, in denen Ihre ETL-Jobs ausgeführt werden, werden mit denselben Sicherheitspraktiken geschützt, denen auch andere AWS Dienste folgen. Einen Überblick über die Verfahren und die gemeinsamen Sicherheitsaufgaben finden Sie im Whitepaper Einführung in AWS Sicherheitsprozesse.

AWS Glue Konzepte

AWS Glue ist ein vollständig verwalteter ETL-Service (Extrahieren, Transformieren, Laden), mit dem Sie Daten problemlos zwischen verschiedenen Datenquellen und Zielen verschieben können. Die wichtigsten Komponenten sind:

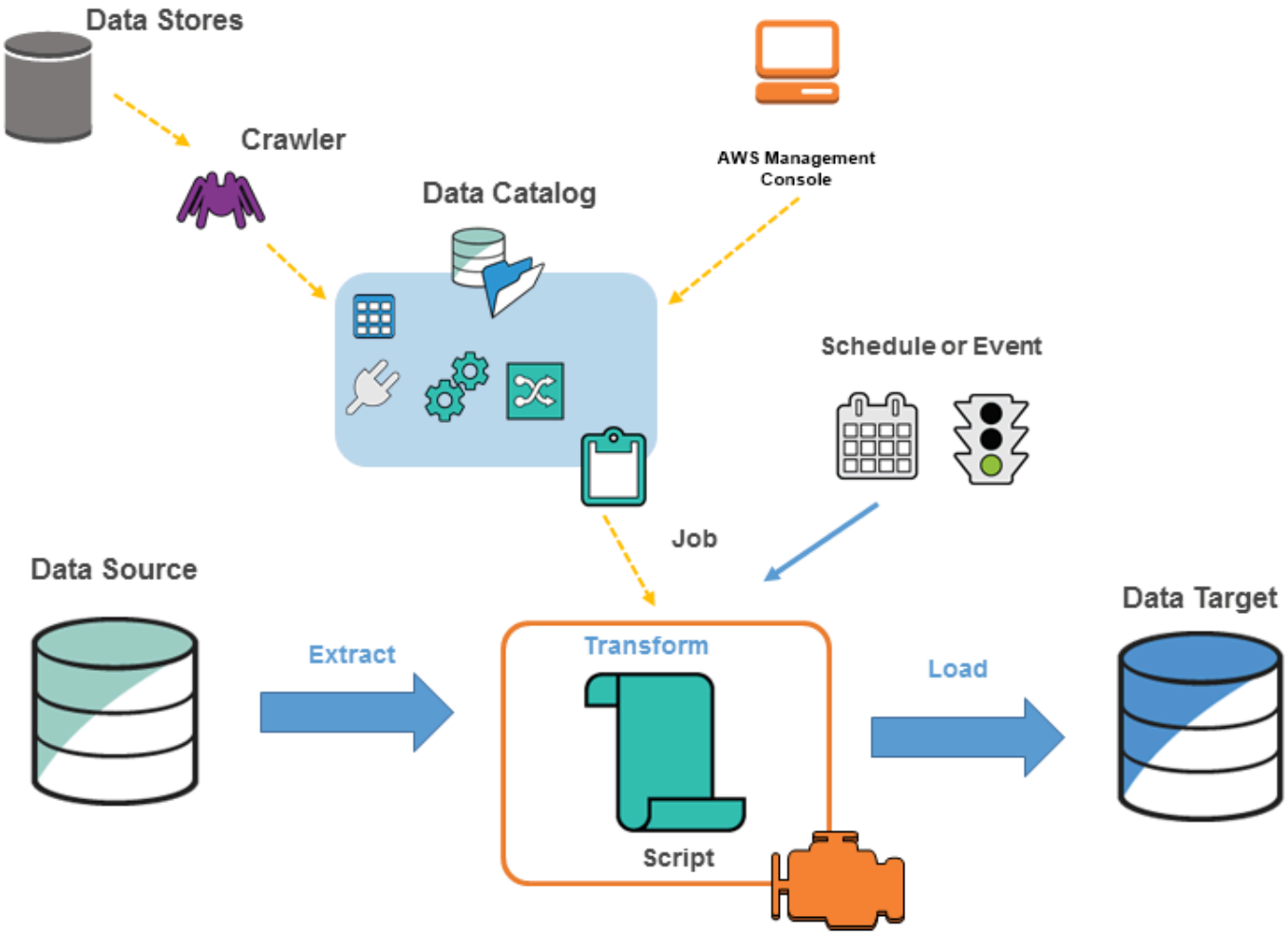
- Datenkatalog: Ein Metadatenspeicher, der Tabellendefinitionen, Jobdefinitionen und andere Kontrollinformationen für Ihre ETL-Workflows enthält.
- Crawler: Programme, die eine Verbindung zu Datenquellen herstellen, Datenschemas ableiten und Metadatentabellendefinitionen im Datenkatalog erstellen.
- ETL-Jobs: Die Geschäftslogik, um Daten aus Quellen zu extrahieren, sie mithilfe von Apache Spark-Skripts zu transformieren und in Ziele zu laden.

- Auslöser: Mechanismen zum Initiieren von Jobausführungen auf der Grundlage von Zeitplänen oder Ereignissen.

Der typische Arbeitsablauf umfasst:

1. Definieren Sie Datenquellen und Ziele im Datenkatalog.
2. Verwenden Sie Crawler, um den Datenkatalog mit Tabellenmetadaten aus Datenquellen zu füllen.
3. Definieren Sie ETL-Jobs mit Transformationsskripten, um Daten zu verschieben und zu verarbeiten.
4. Führen Sie Jobs bei Bedarf oder basierend auf Triggern aus.
5. Überwachen Sie die Auftragsleistung mithilfe von Dashboards.

Das folgende Diagramm zeigt die Architektur einer AWS Glue Umgebung.



Sie definieren Jobs AWS Glue , um die Arbeit zu erledigen, die zum Extrahieren, Transformieren und Laden (ETL) von Daten aus einer Datenquelle in ein Datenziel erforderlich ist. Sie können normalerweise folgende Aktionen ausführen:

- Für Datastore-Quellen definieren Sie einen Crawler, um den AWS Glue Data Catalog mit Metadaten-Tabellendefinitionen zu füllen. Sie verweisen Ihren Crawler auf einen Datenspeicher und der Crawler legt Tabellendefinitionen im Data Catalog an. Für Streaming-Quellen definieren Sie manuell Data-Catalog-Tabellen und legen Eigenschaften für den Datenstrom fest.

  Zusätzlich zu den Tabellendefinitionen AWS Glue Data Catalog enthält der weitere Metadaten, die zur Definition von ETL-Jobs erforderlich sind. Sie verwenden diese Metadaten, wenn Sie eine Aufgabe definieren, um Ihre Daten zu transformieren.

- AWS Glue kann ein Skript zur Transformation Ihrer Daten generieren. Oder Sie können das Skript in der AWS Glue Konsole oder API bereitstellen.
- Sie können Ihre Aufgabe bei Bedarf ausführen oder sie so einrichten, dass sie bei Auftreten eines bestimmten Auslösers gestartet wird. Der Auslöser kann zeitbasiert oder ein Ereignis sein.

  Wenn Ihr Auftrag ausgeführt wird, extrahiert ein Skript die Daten aus Ihrer Datenquelle, transformiert die Daten und lädt sie in Ihr Datenziel. Das Skript wird in einer Apache-Spark-Umgebung in ausgeführt AWS Glue.

> **Important**
>
> Tabellen und Datenbanken in AWS Glue sind Objekte in der AWS Glue Data Catalog. Sie enthalten Metadaten – keine Daten aus einem Datenspeicher.

Textbasierte Daten, wie z. B. CSVs, müssen codiert werden, AWS Glue damit sie **`UTF-8`** erfolgreich verarbeitet werden können. Weitere Informationen finden Sie unter UTF-8 in Wikipedia.

AWS Glue Terminologie

AWS Glue stützt sich bei der Erstellung und Verwaltung Ihres ETL-Workflows (Extrahieren, Transformieren und Laden) auf das Zusammenspiel mehrerer Komponenten.

AWS Glue Data Catalog

Die persistenten Metadaten werden in gespeichert AWS Glue. Es enthält Tabellendefinitionen, Jobdefinitionen und andere Kontrollinformationen zur Verwaltung Ihrer AWS Glue Umgebung. Jedes AWS Konto hat ein Konto AWS Glue Data Catalog pro Region.

Classifier

Bestimmt das Schema Ihrer Daten. AWS Glue bietet Klassifikatoren für gängige Dateitypen wie CSV, JSON, AVRO, XML und andere. Es stellt auch Classifier für gängige relationale Datenbankmanagementsysteme mit einer JDBC-Verbindung zur Verfügung. Sie können einen eigenen Classifier schreiben, indem Sie ein Grok-Muster verwenden oder indem Sie einen Row-Tag in einem XML-Dokument festlegen.

Verbindung

Ein Data-Catalog-Objekt mit den Eigenschaften, die für die Verbindung mit einem bestimmten Datenspeicher erforderlich sind.

Crawler

Ein Programm, das sich mit einem Datenspeicher (Quelle oder Ziel) verbindet, eine priorisierte Liste von Classifiern verarbeitet, um das Schema für Ihre Daten zu bestimmen, und dann Metadatentabellen im AWS Glue Data Catalog erstellt.

Datenbank

Eine Gruppe zugeordneter Data-Catalog-Tabellendefinitionen, die in einer logischen Gruppe organisiert sind.

Datenspeicher, Datenquelle, Datenziel

Ein Datenspeicher ist ein Repository für die dauerhafte Speicherung Ihrer Daten. Beispiele hierfür sind Amazon-S3-Buckets und relationale Datenbanken. Eine Datenquelle ist ein Datenspeicher, der als Eingabe für einen Prozess oder eine Transformation verwendet wird. Ein Datenziel ist ein Datenspeicher, in den ein Prozess oder eine Transformation schreibt.

Entwicklungsendpunkt

Eine Umgebung, die Sie zum Entwickeln und Testen Ihrer AWS Glue ETL-Skripts verwenden können.

Dynamischer Frame

Eine verteilte Tabelle, die verschachtelte Daten wie Strukturen und Arrays unterstützt. Jeder Datensatz ist selbstbeschreibend und wurde auf Schema-Flexibilität mit halbstrukturierten Daten ausgelegt. Jeder Datensatz enthält sowohl Daten als auch das Schema, das diese Daten beschreibt. Sie können sowohl dynamische Frames als auch Apache Spark DataFrames in Ihren ETL-Skripten verwenden und zwischen diesen konvertieren. Dynamische Frames bieten eine Reihe von erweiterten Transformationen für die Datenbereinigung und für ETL.

Aufgabe

Die Geschäftslogik, die für die Ausführung von ETL-Arbeiten erforderlich ist. Sie besteht aus einem Transformationsskript, Datenquellen und Datenzielen. Auftragsausführungen werden durch Auslöser ausgelöst. Diese können geplant sein oder durch Ereignisse ausgelöst werden.

Dashboard zur Auftragsperformance

AWS Glue bietet ein umfassendes Run-Dashboard für Ihre ETL-Jobs. Das Dashboard zeigt Informationen zu Auftragsausführungen in einem bestimmten Zeitraum an.

Notebook-Schnittstelle

Ein verbessertes Notebook-Erlebnis mit Ein-Klick-Einrichtung für einfache Auftragserstellung und Datenexploration. Das Notebook und die Connectors werden automatisch für Sie konfiguriert. Sie können die auf Jupyter Notebook basierende Notebook-Oberfläche verwenden, um Skripts und Workflows mithilfe der AWS Glue serverlosen Apache Spark ETL-Infrastruktur interaktiv zu entwickeln, zu debuggen und bereitzustellen. Sie können auch Ad-hoc-Abfragen, Datenanalysen und Visualisierung (z. B. Tabellen und Diagramme) in der Notebook-Umgebung durchführen.

Script

Code, der Daten aus Quellen extrahiert, transformiert und in Ziele lädt. AWS Glue generiert PySpark oder Scala-Skripte.

Tabelle

Die Metadaten-Definition, die Ihre Daten repräsentiert. Unabhängig davon, ob sich Ihre Daten in einer Amazon Simple Storage Service (Amazon S3)-, einer Amazon Relational Database Service (Amazon RDS)-Tabelle oder anderen Datenelementen befinden, definiert eine Tabelle das Schema Ihrer Daten. Eine Tabelle in der AWS Glue Data Catalog besteht aus den Namen von Spalten, Datentypdefinitionen, Partitionsinformationen und anderen Metadaten zu einem Basisdatensatz. Das Schema Ihrer Daten ist in Ihrer AWS Glue Tabellendefinition dargestellt. Die eigentlichen Daten verbleiben in ihrem ursprünglichen Datenspeicher, unabhängig davon, ob sie sich in einer Datei oder einer relationalen Datenbanktabelle befinden. AWS Glue katalogisiert Ihre Dateien und relationalen Datenbanktabellen in der. AWS Glue Data Catalog Sie werden als Quellen und Ziele verwendet, wenn Sie einen ETL-Auftrag anlegen.

Transform

Die Codelogik, die verwendet wird, um Ihre Daten in ein anderes Format zu bringen.

Auslöser

Initiiert einen ETL-Auftrag. Auslöser können auf der Grundlage einer geplanten Uhrzeit oder eines Ereignisses definiert werden.

Visueller Auftragseditor

Der visuelle Auftrags-Editor ist eine grafische Oberfläche, mit der Sie ETL-Aufträge (Extract, Transform, Load) in AWS Glue ganz einfach erstellen, ausführen und überwachen können. Sie können Workflows zur Datentransformation visuell zusammenstellen, sie nahtlos auf der AWS Glue Apache Spark-basierten serverlosen ETL-Engine ausführen und das Schema und die Datenergebnisse in jedem Schritt des Jobs überprüfen.

Worker

Mit zahlen Sie nur für die Zeit AWS Glue, die Ihr ETL-Job für die Ausführung benötigt. Sie müssen keine Ressourcen verwalten, es gibt keine Vorabkosten und Ihnen werden keine Start- oder Shutdown-Zeit in Rechnung gestellt. Ihnen wird ein Stundensatz berechnet, der auf der Anzahl der Datenverarbeitungseinheiten (oder DPUs) basiert, die für die Ausführung Ihres ETL-Jobs verwendet werden. Eine einzelne Datenverarbeitungseinheit (DPU) wird auch als Worker bezeichnet. AWS Glue verfügt über mehrere Workertypen, sodass Sie die Konfiguration auswählen können, die Ihren Anforderungen an die Auftragslatenz und die Kosten entspricht. Worker sind in Standard-, G.1X-, G.2X-, G.4X-, G.8X-, G.12X-, G.16X-, G.025X- und speicheroptimierten Konfigurationen R.1X, R.2X, R.4X, R.8X erhältlich.

AWS Glue Komponenten

AWS Glue bietet eine Konsole und API-Operationen zum Einrichten und Verwalten Ihres ETL-Workloads (Extrahieren, Transformieren und Laden). Sie können API-Operationen über mehrere sprachspezifische Funktionen SDKs und die AWS Command Line Interface ()AWS CLI verwenden. Informationen zur Verwendung von finden Sie unter AWS CLI Befehlsreferenz. AWS CLI

AWS Glue verwendet die AWS Glue Data Catalog , um Metadaten zu Datenquellen, Transformationen und Zielen zu speichern. Der Data Catalog ist ein Drop-In-Ersatz für den Apache-Hive-Metastore. Das AWS Glue Jobs system bietet eine verwaltete Infrastruktur für die Definition, Planung und Ausführung von ETL-Vorgängen für Ihre Daten. Weitere Informationen zur AWS Glue API finden Sie unterAWS Glue API.

AWS Glue Konsole

Sie verwenden die AWS Glue Konsole, um Ihren ETL-Workflow zu definieren und zu orchestrieren. Die Konsole ruft mehrere API-Operationen im AWS Glue Data Catalog und AWS Glue Jobs system auf, um die folgenden Aufgaben auszuführen:

- Definieren Sie AWS Glue Objekte wie Jobs, Tabellen, Crawler und Verbindungen.
- Planen, wann Crawler ausgeführt werden.
- Definieren von Ereignissen oder Zeitplänen für Auftragsauslöser.
- Suchen und filtern Sie AWS Glue Objektlisten.
- Bearbeiten von Transformationskripts.

AWS Glue Data Catalog

Das AWS Glue Data Catalog ist Ihr persistenter Speicher für technische Metadaten in der AWS Cloud.

Jedes AWS Konto hat eines AWS Glue Data Catalog pro AWS Region. Jeder Datenkatalog ist eine hoch skalierbare Sammlung von Tabellen, die in Datenbanken organisiert sind. Eine Tabelle ist eine Metadatendarstellung einer Sammlung strukturierter oder halbstrukturierter Daten, die in Quellen wie Amazon RDS, Apache Hadoop Distributed File System, Amazon OpenSearch Service und anderen gespeichert sind. Sie AWS Glue Data Catalog bietet ein einheitliches Repository, in dem unterschiedliche Systeme Metadaten speichern und finden können, um den Überblick über Daten in Datensilos zu behalten. Sie können die Metadaten dann verwenden, um diese Daten in einer Vielzahl von Anwendungen konsistent abzufragen und zu transformieren.

Sie verwenden den Datenkatalog zusammen mit AWS Identity and Access Management Richtlinien und Lake Formation, um den Zugriff auf die Tabellen und Datenbanken zu kontrollieren. Auf diese Weise können Sie verschiedenen Gruppen in Ihrem Unternehmen erlauben, Daten sicher für die gesamte Organisation zu veröffentlichen, während vertrauliche Informationen auf sehr granulare Weise geschützt werden.

Der Datenkatalog bietet Ihnen zusammen mit CloudTrail Lake Formation auch umfassende Audit- und Governance-Funktionen mit Nachverfolgung von Schemaänderungen und Datenzugriffskontrollen. Auf diese Weise stellen Sie sicher, dass Daten nicht unangemessen geändert oder versehentlich freigegeben werden.

Informationen zum Sichern und Prüfen finden Sie unter AWS Glue Data Catalog:

- AWS Lake Formation – Weitere Informationen finden Sie unter Was ist AWS Lake Formation? im AWS Lake Formation -Entwicklerhandbuch.
- CloudTrail— Weitere Informationen finden Sie unter Was ist CloudTrail? im AWS CloudTrail Benutzerhandbuch.

Im Folgenden finden Sie weitere AWS Dienste und Open-Source-Projekte, die Folgendes verwenden: AWS Glue Data Catalog

- Amazon Athena – Weitere Informationen finden Sie unter Grundlegendes zu Tabellen, Datenbanken und zum Data Catalog im Amazon-Athena-Benutzerhandbuch.
- Amazon Redshift Spectrum – Weitere Informationen finden Sie unter Verwenden von Amazon Redshift Spectrum zum Abfragen externer Daten im Datenbank-Entwicklerhandbuch für Amazon Redshift.
- Amazon EMR – Weitere Informationen finden Sie unter Verwenden von ressourcenbasierten Richtlinien für Amazon-EMR-Zugriff auf AWS Glue Data Catalog im Amazon-EMR-Managementhandbuch.
- AWS Glue Data Catalog Client für Apache Hive Metastore — Weitere Informationen zu diesem GitHub Projekt finden Sie unter AWS Glue Data Catalog Client für Apache Hive Metastore.

AWS Glue Crawler und Klassifikatoren

AWS Glue ermöglicht es Ihnen auch, Crawler einzurichten, die Daten in allen Arten von Repositorys scannen, klassifizieren, Schemainformationen daraus extrahieren und die Metadaten automatisch in der speichern können. AWS Glue Data Catalog AWS Glue Data Catalog Sie können dann zur Steuerung von ETL-Vorgängen verwendet werden.

Informationen dazu, wie Crawler und Classifier eingerichtet werden, finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs . Informationen zur Programmierung von Crawlern und Classifiern mithilfe der AWS Glue API finden Sie unter. Crawler- und Classifier-API

AWS Glue ETL-Operationen

Mithilfe der Metadaten im Datenkatalog AWS Glue können automatisch Scala- oder PySpark (die Python-API für Apache Spark) -Skripts mit AWS Glue Erweiterungen generiert werden, die Sie verwenden und ändern können, um verschiedene ETL-Operationen auszuführen. Sie können

beispielsweise Rohdaten extrahieren, bereinigen und umwandeln und das Ergebnis anschließend in einem anderen Repository speichern, wo es abgefragt und analysiert werden kann. Ein solches Skript könnte eine CSV-Datei in ein relationales Format konvertieren und in Amazon Redshift speichern.

Weitere Informationen zur Verwendung von AWS Glue ETL-Funktionen finden Sie unterProgrammieren von Spark-Skripte.

ETL einstreamen AWS Glue

AWS Glue ermöglicht es Ihnen, ETL-Operationen mit Streaming-Daten mithilfe kontinuierlich ausgeführter Jobs durchzuführen. AWS Glue Streaming-ETL basiert auf der Apache Spark Structured Streaming Engine und kann Streams von Amazon Kinesis Data Streams, Apache Kafka und Amazon Managed Streaming for Apache Kafka (Amazon MSK) aufnehmen. Streaming-ETL kann Streaming-Daten bereinigen und transformieren und in Amazon-S3- oder JDBC-Datastores laden. Verwenden Sie Streaming ETL in AWS Glue , um Ereignisdaten wie IoT-Streams, Clickstreams und Netzwerkprotokolle zu verarbeiten.

Wenn Sie das Schema der Streaming-Datenquelle kennen, können Sie es in einer Data-Catalog-Tabelle angeben. Andernfalls können Sie die Schemaerkennung im Streaming-ETL-Auftrag aktivieren. Der Auftrag bestimmt dann automatisch das Schema aus den eingehenden Daten.

Der Streaming-ETL-Job kann sowohl AWS Glue integrierte Transformationen als auch Transformationen verwenden, die systemeigenen Transformationen von Apache Spark Structured Streaming sind. Weitere Informationen finden Sie unter Operations on streaming DataFrames / Datasets auf der Apache Spark-Website.

Weitere Informationen finden Sie unter the section called “Streaming-ETL-Aufträge”.

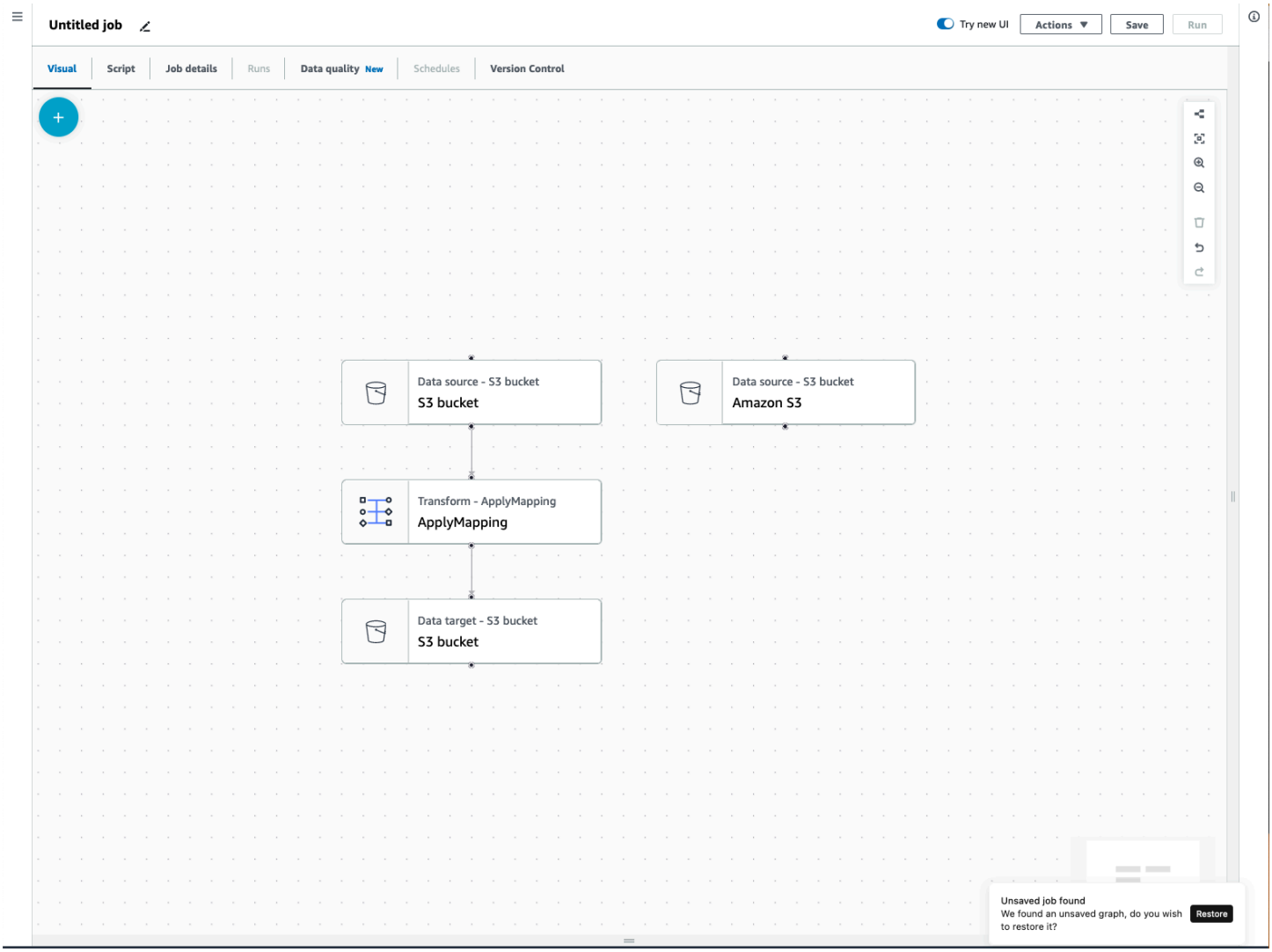
Das Jobsystem AWS Glue

Das AWS Glue Jobs system bietet eine verwaltete Infrastruktur zur Orchestrierung Ihres ETL-Workflows. Sie können Jobs erstellen AWS Glue , die die Skripts automatisieren, die Sie zum Extrahieren, Transformieren und Übertragen von Daten an verschiedene Speicherorte verwenden. Aufträge können zeitlich geplant und verkettet werden. Oder sie können von Ereignissen wie etwa dem Eintreffen neuer Daten ausgelöst werden.

Weitere Informationen zur Verwendung von finden Sie unterÜberwachung AWS Glue. AWS Glue Jobs system Informationen zum Programmieren mithilfe der AWS Glue Jobs system -API finden Sie unter Auftrags-API.

Visuelle ETL-Komponenten

AWS Mit Glue können Sie ETL-Jobs über eine visuelle Leinwand erstellen, die Sie bearbeiten können.



Menü ETL-Auftrag

Über die Menüoptionen oben im Zeichenbereich können Sie auf die verschiedenen Ansichten und Konfigurationsdetails zu Ihrem Auftrag zugreifen.

- Visuell – Der Zeichenbereich des visuellen Auftrags-Editors. Hier können Sie Knoten hinzufügen, um einen Auftrag zu erstellen.
- Script — Die Skriptdarstellung Ihres ETL-Jobs. AWS Glue generiert das Skript auf der Grundlage der visuellen Darstellung Ihres Jobs. Sie können Ihr Skript auch bearbeiten oder es herunterladen.

> **Note**
>
> Bei Auswahl der Skriptbearbeitung wird die Auftragserstellung dauerhaft in einen reinen Skriptmodus umgewandelt. Danach können Sie den visuellen Editor nicht mehr für die Bearbeitung des Auftrags verwenden. Bevor Sie das Skript auswählen, sollten Sie alle Auftragsquellen, Transformationen und Ziele hinzufügen und alle gewünschten Änderungen mit dem visuellen Editor vornehmen.

- Auftragsdetails – Auf der Registerkarte Auftragsdetails können Sie Ihren Auftrag konfigurieren, indem Sie die Auftragseigenschaften festlegen. Es gibt grundlegende Eigenschaften wie Name und Beschreibung Ihres Jobs, IAM-Rolle, Jobtyp, AWS Glue-Version, Sprache, Workertyp, Anzahl der Worker, Job-Lesezeichen, Flex-Ausführung, Anzahl der Pensionierungen und Job-Timeout, und es gibt erweiterte Eigenschaften wie Verbindungen, Bibliotheken, Jobparameter und Tags.
- Ausführungen – Nachdem Ihr Auftrag ausgeführt wurde, können Sie auf diese Registerkarte zugreifen, um Ihre vergangenen Auftragsausführungen anzuzeigen.
- Datenqualität – Die Datenqualität bewertet und überwacht die Qualität Ihrer Datenbestände. Auf dieser Registerkarte erfahren Sie mehr darüber, wie Sie die Datenqualität verwenden und eine Datenqualitätstransformation zu Ihrem Auftrag hinzufügen können.
- Zeitpläne – Von Ihnen geplante Aufträge werden auf dieser Registerkarte angezeigt. Wenn diesem Auftrag keine Zeitpläne angefügt sind, dann ist diese Registerkarte nicht zugänglich.
- Versionskontrolle – Sie können Git mit Ihrem Auftrag verwenden, indem Sie Ihren Auftrag für ein Git-Repository konfigurieren.

Visuelle ETL-Bedienfelder

Wenn Sie im Zeichenbereich arbeiten, stehen Ihnen mehrere Bedienfelder zur Verfügung, die Sie bei der Konfiguration Ihrer Knoten unterstützen oder Ihnen dabei helfen, eine Vorschau Ihrer Daten anzuzeigen und das Ausgabeschema anzuzeigen.

- Eigenschaften – Das Bedienfeld Eigenschaften wird angezeigt, wenn Sie einen Knoten in Ihrem Zeichenbereich auswählen.
- Datenvorschau – Das Bedienfeld Datenvorschau bietet eine Vorschau der Datenausgabe, so dass Sie Entscheidungen treffen können, bevor Sie Ihren Auftrag ausführen und Ihre Ausgabe prüfen.
- Ausgabeschema – Auf der Registerkarte Ausgabeschema können Sie das Schema Ihrer Transformationsknoten anzeigen und bearbeiten.

Größe der Bedienfelder ändern

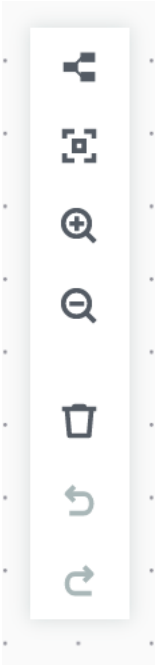
Sie können die Größe des Eigenschaftenbereichs auf der rechten Seite des Bildschirms und des unteren Bereichs, der die Registerkarten Datenvorschau und Ausgabeschema enthält, ändern, indem Sie auf den Edge des Bereichs klicken und ihn nach links und rechts oder nach oben und unten ziehen.

- Eigenschaftenfenster – Ändern Sie die Größe des Eigenschaftenfensters, indem Sie auf den Edge des Zeichenbereichs auf der rechten Seite des Bildschirms klicken und ihn ziehen. Ziehen Sie ihn dann nach links, um seine Breite zu vergrößern. Standardmäßig ist das Bedienfeld reduziert und wenn ein Knoten ausgewählt wird, wird das Eigenschaftenfeld in seiner Standardgröße geöffnet.
- Datenvorschau und Bedienfeld Ausgabeschema – Ändern Sie die Größe des unteren Bedienfelds, indem Sie auf den unteren Edge des Zeichenbereichs am unteren Bildschirmrand klicken und ihn nach oben ziehen, um seine Höhe zu vergrößern. Standardmäßig ist das Bedienfeld reduziert und wenn ein Knoten ausgewählt wird, wird das untere Bedienfeld in seiner Standardgröße geöffnet.

Zeichenbereich für Aufträge

Sie können Knoten direkt im Zeichenbereich für visuelle ETLs hinzufügen, entfernen und verschieben/neu anordnen. Betrachten Sie es als Ihren Workspace zur Erstellung eines voll funktionsfähigen ETL-Auftrags, der mit einer Datenquelle beginnt und mit einem Datenziel enden kann.

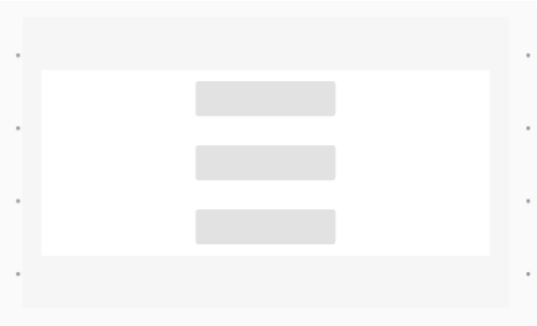
Wenn Sie mit Knoten auf dem Zeichenbereich arbeiten, steht Ihnen eine Symbolleiste zur Verfügung, mit der Sie hinein- und herauszoomen, Knoten entfernen, Verbindungen zwischen Knoten herstellen oder bearbeiten, die Auftragsflussausrichtung ändern und eine Aktion rückgängig machen oder wiederholen können.



Die schwebende Symbolleiste ist in der oberen rechten Ecke des Zeichenbereichs verankert und enthält mehrere Images, die Aktionen ausführen:

- Layout-Symbol – Das erste Symbol in der Symbolleiste ist das Layout-Symbol. In der Standardeinstellung ist die Richtung der visuellen Aufträge von oben nach unten gerichtet. Es ändert die Richtung Ihres visuellen Auftrags, indem es die Knoten horizontal von links nach rechts anordnet. Durch erneutes Klicken auf das Layout-Symbol ändert sich die Richtung wieder von oben nach unten.
- Symbol „Neu zentrieren“ – Das Symbol „Neu zentrieren“ ändert die Zeichenbereichsansicht, indem es sie zentriert. Sie können dies bei großen Aufträgen verwenden, um wieder in die Mittelstellung zu gelangen.
- Symbol „Vergrößern“ – Das Symbol „Vergrößern“ vergrößert die Größe der Knoten im Zeichenbereich.
- Symbol „Verkleinern“ – Das Symbol „Verkleinern“ verringert die Größe der Knoten im Zeichenbereich.
- Papierkorbsymbol – Mit dem Papierkorbsymbol entfernen Sie einen Knoten aus dem visuellen Auftrag. Sie müssen zuerst einen Knoten auswählen.
- Symbol „Rückgängig“ – Das Rückgängig-Symbol macht die letzte am visuellen Auftrag ausgeführte Aktion rückgängig.
- Symbol „Wiederherstellen“ – Das Wiederherstellen-Symbol wiederholt die letzte Aktion, die für den visuellen Auftrag ausgeführt wurde.

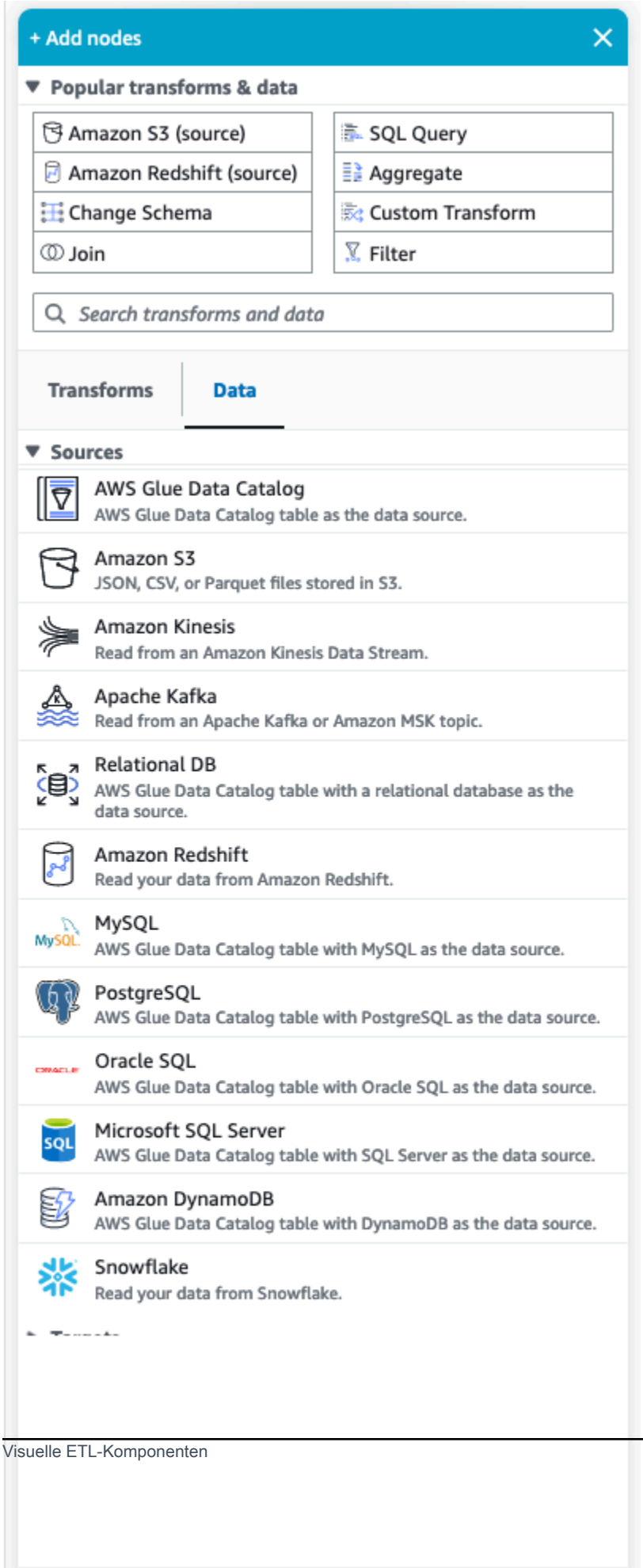
Verwendung der Minikarte



Bedienfeld Ressourcen

Der Ressourcenbereich enthält alle für Sie verfügbaren Datenquellen, Transformationsaktionen und Verbindungen. Öffnen Sie den Ressourcenbereich im Zeichenbereich, indem Sie auf das „+“-Symbol klicken. Dadurch wird der Ressourcenbereich geöffnet.

Klicken Sie zum Schließen den Ressourcenbereich auf das X in der oberen rechten Ecke des Ressourcenbereichs. Dadurch wird das Bedienfeld ausgeblendet, bis Sie es erneut öffnen können.


+ Add nodes
Popular transforms & data
Amazon S3 (source)
Amazon Redshift (source)
Change Schema
Join
SQL Query
Aggregate
Custom Transform
Filter
Search transforms and data
Transforms
Data
Sources
AWS Glue Data Catalog
AWS Glue Data Catalog table as the data source.
Amazon S3
JSON, CSV, or Parquet files stored in S3.
Amazon Kinesis
Read from an Amazon Kinesis Data Stream.
Apache Kafka
Read from an Apache Kafka or Amazon MSK topic.
Relational DB
AWS Glue Data Catalog table with a relational database as the data source.
Amazon Redshift
Read your data from Amazon Redshift.
MySQL
AWS Glue Data Catalog table with MySQL as the data source.
PostgreSQL
AWS Glue Data Catalog table with PostgreSQL as the data source.
Oracle SQL
AWS Glue Data Catalog table with Oracle SQL as the data source.
Microsoft SQL Server
AWS Glue Data Catalog table with SQL Server as the data source.
Amazon DynamoDB
AWS Glue Data Catalog table with DynamoDB as the data source.
Snowflake
Read your data from Snowflake.

Beliebte Transformationen und Daten

Im oberen Bereich des Bedienfelds befindet sich eine Sammlung beliebter Transformationen und Daten. Diese Knoten werden häufig in AWS Glue verwendet. Wählen Sie eines aus, um es dem Zeichenbereich hinzuzufügen. Sie können die beliebten Transformationen und Daten auch ausblenden, indem Sie auf das Dreieck neben der Überschrift Beliebte Transformationen und Daten klicken.

Unter dem Abschnitt Beliebte Transformationen und Daten können Sie nach Transformationen und Datenquellenknoten suchen. Die Ergebnisse werden während der Eingabe angezeigt. Je mehr Buchstaben Sie zu Ihrer Suchanfrage hinzufügen, desto kleiner wird die Ergebnisliste. Suchergebnisse werden anhand des Knotennamens und/oder der Beschreibung aufgefüllt. Wählen Sie den Knoten aus, um ihn Ihrem Zeichenbereich hinzuzufügen.

Transformationen und Daten

Es gibt zwei Registerkarten, auf denen die Knoten in Transformationen und Daten unterteilt sind.

Transformationen – Wenn Sie die Registerkarte Transformationen auswählen, können alle verfügbaren Transformationen ausgewählt werden. Wählen Sie eine Transformation aus, um sie dem Zeichenbereich hinzuzufügen. Sie können auch die Option Transformation hinzufügen unten in der Liste Transformationen auswählen, um eine neue Seite mit der Dokumentation zum Erstellen benutzerdefinierter visueller Transformationen zu öffnen. Wenn Sie die Schritte befolgen, können Sie eigene Transformationen erstellen. Ihre Transformationen werden dann in der Liste der verfügbaren Transformationen angezeigt.

Daten – Die Registerkarte Daten enthält alle Knoten für Quellen und Ziele. Sie können die Quellen und Ziele ausblenden, indem Sie auf das Dreieck neben der Überschrift Quellen oder Ziele klicken. Sie können die Quellen und Ziele wieder einblenden, indem Sie erneut auf das Dreieck klicken. Wählen Sie einen Quell- oder Zielknoten aus, um ihn dem Zeichenbereich hinzuzufügen. Sie können auch Verbindungen verwalten auswählen, um eine neue Verbindung hinzuzufügen. Dadurch wird die Seite Konnektoren in der Konsole geöffnet.

AWS Glue für Spark und AWS Glue für Ray

In AWS Glue Apache Spark (AWS Glue ETL) können PySpark Sie Python-Code schreiben, um Daten in großem Maßstab zu verarbeiten. Spark ist eine vertraute Lösung für dieses Problem, aber Dateningenieure mit einem Python-fokussierten Hintergrund können den Übergang als unintuitiv

empfinden. Das DataFrame Spark-Modell ist nicht nahtlos „pythonisch“, was die Scala-Sprache und die Java-Laufzeit widerspiegelt, auf der es aufbaut.

In AWS Glue können Sie Python-Shell-Jobs verwenden, um native Python-Datenintegrationen auszuführen. Diese Jobs werden auf einer einzigen EC2 Amazon-Instance ausgeführt und sind durch die Kapazität dieser Instance begrenzt. Dies schränkt den Datendurchsatz ein, den Sie verarbeiten können, und wird bei Big Data teuer in der Wartung.

AWS Glue for Ray ermöglicht es Ihnen, Python-Workloads zu skalieren, ohne nennenswerte Investitionen in das Erlernen von Spark zu tätigen. Sie können bestimmte Szenarien nutzen, in denen Ray eine bessere Leistung erbringt. Indem wir Ihnen eine Auswahl anbieten, können Sie die Stärken von Spark und Ray nutzen.

AWS Glue ETL und AWS Glue for Ray unterscheiden sich grundlegend, sodass sie unterschiedliche Funktionen unterstützen. Bitte prüfen Sie die Dokumentation, um die unterstützten Features zu bestimmen.

Was ist AWS Glue für Ray?

Ray ist ein Open-Source-Framework für verteilte Berechnungen, mit dem Sie Workloads mit Schwerpunkt auf Python skalieren können. Weitere Informationen zu Ray finden Sie auf der Website von Ray. AWS Glue Ray-Jobs und interaktive Sessions ermöglichen es Ihnen, Ray intern zu verwenden AWS Glue.

Sie können AWS Glue for Ray verwenden, um Python-Skripte für Berechnungen zu schreiben, die parallel auf mehreren Computern ausgeführt werden. In Ray-Aufträgen und interaktiven Sitzungen können Sie vertraute Python-Bibliotheken wie Pandas verwenden, um Ihre Workflows einfach zu schreiben und auszuführen. Weitere Informationen zu Ray-Datensätzen finden Sie unter Ray-Datensätze in der Ray-Dokumentation. Weitere Informationen über Pandas finden Sie auf der Pandas-Website.

Wenn Sie AWS Glue for Ray verwenden, können Sie Ihre Pandas-Workflows mit Big Data auf Unternehmensebene ausführen — mit nur wenigen Codezeilen. Sie können einen Ray-Job von der AWS Glue Konsole oder dem SDK aus erstellen. AWS Sie können auch eine AWS Glue interaktive Sitzung öffnen, um Ihren Code in einer serverlosen Ray-Umgebung auszuführen. Visuelle Jobs in AWS Glue Studio werden noch nicht unterstützt.

AWS Glue for Ray-Jobs ermöglichen es Ihnen, ein Skript nach einem Zeitplan oder als Reaktion auf ein Ereignis von Amazon auszuführen EventBridge. Jobs speichern Protokollinformationen und

Überwachungsstatistiken CloudWatch , anhand derer Sie den Zustand und die Zuverlässigkeit Ihres Skripts nachvollziehen können. Weitere Informationen zum AWS Glue Jobsystem finden Sie unterthe section called “Arbeiten mit Ray-Aufträgen”.

Ray automatisiert die Skalierung von Python-Code, indem es die Verarbeitung auf einen Cluster von Computern verteilt, die es je nach Auslastung in Echtzeit neu konfiguriert. Dies kann bei bestimmten Workloads zu einer verbesserten Leistung pro Dollar führen. Bei Ray-Jobs haben wir Auto Scaling nativ in das AWS Glue Jobmodell integriert, sodass Sie diese Funktion voll ausnutzen können. Ray-Jobs werden auf AWS Graviton ausgeführt, was insgesamt zu einem höheren Preis-Leistungs-Verhältnis führt.

Zusätzlich zu den Einsparungen können Sie das native Auto Scaling verwenden, um Ray-Workloads auszuführen, ohne Zeit in die Cluster-Wartung, -Abstimmung und -Verwaltung zu investieren. Sie können vertraute, sofort einsatzbereite Open-Source-Bibliotheken wie Pandas und das AWS SDK für Pandas verwenden. Diese verbessern die Iterationsgeschwindigkeit, während Sie auf AWS Glue für Ray entwickeln. Wenn Sie AWS Glue for Ray verwenden, können Sie schnell kostengünstige Datenintegrationsworkloads entwickeln und ausführen.

Umwandlung von halbstrukturierten Schemas in relationale Schemas mit AWS Glue

Es ist üblich, semistrukturierte Daten in relationale Tabellen umzuwandeln. Konzeptionell betrachtet reduzieren Sie ein hierarchisches Schema auf ein relationales Schema. AWS Glue kann diese Konvertierung für Sie durchführen. on-the-fly

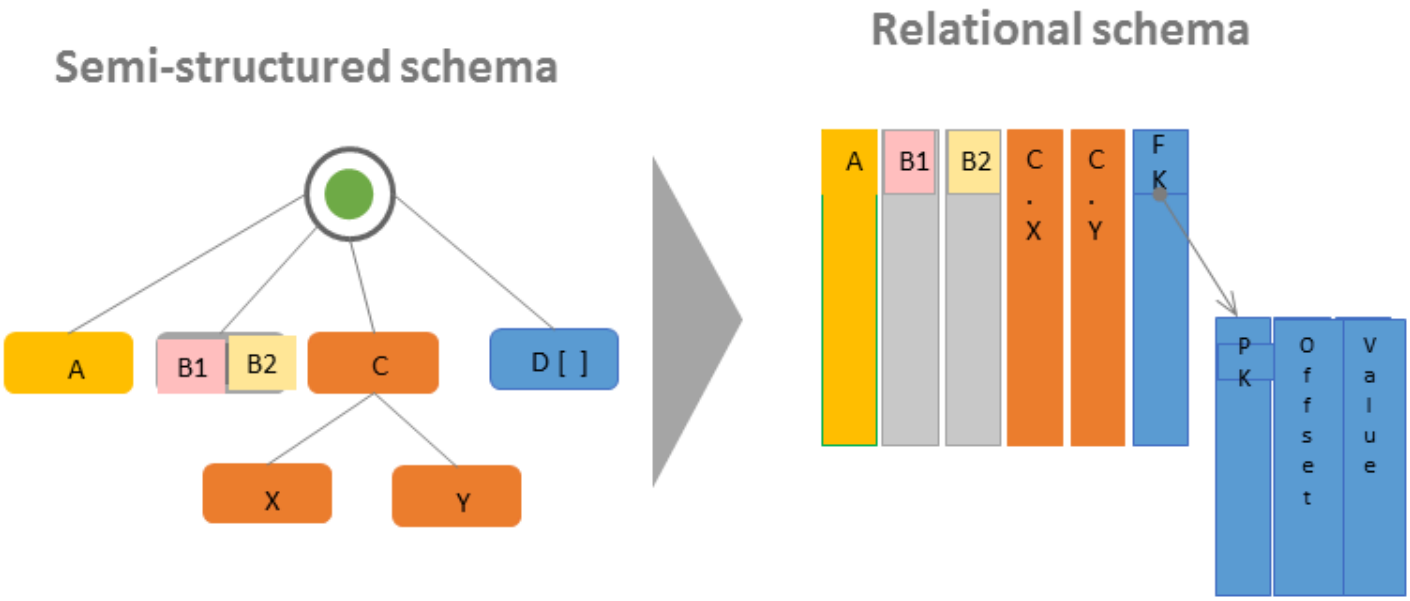
Semistrukturierte Daten enthalten in der Regel Markups zur Identifizierung von Entitäten innerhalb der Daten. Sie können verschachtelte Datenstrukturen ohne festes Schema umfassen. Weitere Informationen zu semistrukturierten Daten finden Sie im Wikipedia-Artikel zu semistrukturierten Daten.

Relationale Daten werden durch Tabellen repräsentiert, die aus Zeilen und Spalten bestehen. Die Beziehungen zwischen Tabellen können durch eine Beziehung zwischen Primärschlüssel (PK) und Fremdschlüssel (FK) dargestellt werden. Weitere Informationen finden Sie im Wikipedia-Artikel zu relationalen Datenbanken.

AWS Glue verwendet Crawler, um Schemas für halbstrukturierte Daten abzuleiten. Es wandelt die Daten dann mit einem ETL-Auftrag (Extract, Transform and Load) in ein relationales Schema um.

Sie können beispielsweise JSON-Daten von Quelldateien aus Amazon Simple Storage Service (Amazon S3) in Amazon Relational Database Service (Amazon RDS)-Tabellen parsen. Verstehen wie AWS Glue Der Umgang mit den Unterschieden zwischen Schemas kann Ihnen helfen, den Transformationsprozess zu verstehen.

Dieses Diagramm zeigt, wie AWS Glue wandelt ein halbstrukturiertes Schema in ein relationales Schema um.



Das Diagramm veranschaulicht folgende Vorgänge:

- Der Einzelwert `A` wird direkt in eine relationale Spalte umgewandelt.
- Das Wertepaar `B1` und `B2` wird in zwei relationale Spalten umgewandelt.
- Die Struktur `C`, mit den untergeordneten Elementen `X` und `Y`, wird in zwei relationale Spalten umgewandelt.

- Array `D[]` wird in eine relationale Spalte mit einem Fremdschlüssel (FK) umgewandelt, der auf eine andere relationale Tabelle verweist. Zusammen mit einem Primärschlüssel (PK) verfügt die zweite relationale Tabelle über Spalten, die den Offset und Wert der Elemente im Array enthalten.

AWS Systeme vom Typ Glue

AWS Glue verwendet Systeme verschiedener Typen, um eine vielseitige Schnittstelle zu Datensystemen bereitzustellen, die Daten auf sehr unterschiedliche Weise speichern. In diesem Dokument werden Systeme und Datenstandards vom Typ AWS Glue unterschieden.

AWS Typen von Glue-Datenkatalogen

Der Datenkatalog ist ein Register von Tabellen und Feldern, die in verschiedenen Datensystemen gespeichert sind, ein Metaspeicher. Wenn AWS Glue-Komponenten wie AWS Glue-Crawler und AWS Glue with Spark-Jobs in den Datenkatalog schreiben, tun sie dies mit einem internen Typsystem zur Nachverfolgung der Feldtypen. Diese Werte werden in der Spalte Datentyp des Tabellenschemas in der AWS Glue-Konsole angezeigt. Dieses Typsystem basiert auf dem Typsystem von Apache Hive. Weitere Informationen zum Apache-Hive-Typsystem finden Sie unter Typen im Apache-Hive-Wiki. Weitere Informationen zu bestimmten Typen und Unterstützung finden Sie in der AWS Glue Console als Teil des Schema Builders anhand von Beispielen.

Validierung, Kompatibilität und andere Verwendungen

Der Datenkatalog validiert keine Typen, die in Typfelder geschrieben wurden. Wenn AWS Glue-Komponenten den Datenkatalog lesen und in ihn schreiben, sind sie miteinander kompatibel. AWS Die Klebstoffkomponenten zielen auch darauf ab, ein hohes Maß an Kompatibilität mit den Hive-Typen zu gewährleisten. AWS Glue-Komponenten garantieren jedoch nicht die Kompatibilität mit allen Hive-Typen. Dies ermöglicht die Interoperabilität mit Tools wie Athena DDL bei der Arbeit mit Tabellen im Datenkatalog.

Da der Datenkatalog keine Typen validiert, können andere Services den Datenkatalog verwenden, um Typen mithilfe von Systemen zu verfolgen, die strikt dem Hive-Typsystem oder jedem anderen System entsprechen.

Typen in AWS Glue mit Spark-Skripten

Wenn ein AWS Glue with Spark-Skript einen Datensatz interpretiert oder transformiert, stellen wir eine speicherinterne Darstellung Ihres Datensatzes bereit`DynamicFrame`, so wie er in Ihrem

Skript verwendet wird. Das Ziel von einem `DynamicFrame` ähnelt dem von Spark `DataFrame` – es modelliert Ihren Datensatz so, dass Spark Transformationen für Ihre Daten planen und ausführen kann. Wir garantieren, dass die Typdarstellung von `DynamicFrame` kompatibel mit `DataFrame` ist, indem wir die `toDF`- und `fromDF`-Methoden bereitstellen.

Wenn Typinformationen für einen `DataFrame` abgeleitet oder bereitgestellt werden können, können sie auch für einen `DynamicFrame` abgeleitet oder bereitgestellt werden, sofern nicht anders dokumentiert. Wenn wir optimierte Lese- oder Schreibprogramme für bestimmte Datenformate bereitstellen, können die von Spark bereitgestellten Lese- und Schreibprogramme Ihre Daten lesen oder schreiben, sofern dies mit den dokumentierten Einschränkungen möglich ist. Weitere Informationen über Lese- oder Schreibprogramme finden Sie unter the section called “Pfad-Formatoptionen”.

Der Typ der Wahl

`DynamicFrames` stellen einen Mechanismus zur Modellierung von Feldern in einem Datensatz bereit, deren Werte auf der Festplatte zeilenübergreifend inkonsistente Typen haben können. Beispielsweise kann ein Feld eine Zahl enthalten, die in bestimmten Zeilen als Zeichenfolge gespeichert ist, und in anderen eine Ganzzahl. Dieser Mechanismus ist ein In-Memory-Typ, bezeichnet als `Choice`. Wir bieten Transformationen wie die `ResolveChoice` Methode an, um Choice-Spalten in einen konkreten Typ aufzulösen. AWS Glue ETL schreibt den Choice-Typ im normalen Betrieb nicht in den Datenkatalog. Choice-Typen existieren nur im Kontext von DynamicFrame Speichermodellen von Datensätzen. Ein Beispiel für die Verwendung des Choice-Typs finden Sie unter the section called “Beispiel zu Datenvorbereitung”.

AWS Typen von Glue Crawler

Crawler zielen darauf ab, ein konsistentes, verwendbares Schema für Ihren Datensatz zu erstellen und es dann im Datenkatalog zu speichern, um es in anderen AWS Glue-Komponenten und Athena zu verwenden. Crawler arbeiten mit Typen, wie im vorherigen Abschnitt über den Datenkatalog beschrieben, the section called “AWS Typen von Glue-Datenkatalogen”. Um einen verwendbaren Typ in Szenarien vom Typ „Auswahl“ zu erzeugen, in denen eine Spalte Werte von zwei oder mehr Typen enthält, erstellen Crawler einen `struct`-Typ, der die potenziellen Typen modelliert.

Erste Schritte mit AWS Glue

Die folgenden Abschnitte enthalten Informationen zur Einrichtung AWS GlueNicht alle Einrichtungsabschnitte sind erforderlich, um mit der Verwendung von zu beginnen. AWS Glue. Sie können die Anweisungen nach Bedarf verwenden, um IAM-Berechtigungen, Verschlüsselung und DNS einzurichten (wenn Sie eine VPC-Umgebung für den Zugriff auf Datenspeicher verwenden oder interaktive Sitzungen verwenden).

Themen

- Überblick über die Verwendung AWS Glue
- Einrichten von IAM-Berechtigungen für AWS Glue
- AWS Glue Nutzungsprofile einrichten
- Erste Schritte mit dem AWS Glue Data Catalog
- Netzwerkzugriff auf Datenspeicher einrichten
- Einrichten der Verschlüsselung in AWS Glue
- Einrichtung von Netzwerken für die Entwicklung für AWS Glue

Überblick über die Verwendung AWS Glue

Mit AWS Glue speichern Sie Metadaten in der AWS Glue Data Catalog. Sie verwenden diese Metadaten, um ETL-Aufträge zu steuern, die Datenquellen transformieren und Ihr Data Warehouse oder Data Lake laden. In den folgenden Schritten werden der allgemeine Arbeitsablauf und einige der Optionen beschrieben, die Sie bei der Arbeit mit AWS Glue.

> **Note**
>
> Sie können die folgenden Schritte ausführen oder einen Workflow erstellen, der die Schritte 1 bis 3 automatisch ausführt. Weitere Informationen finden Sie unter the section called “Ausführen von komplexen ETL-Aktivitäten mithilfe von Blueprints und Workflows”.

1. Füllen Sie die AWS Glue Data Catalog mit Tabellendefinitionen aus.

   In der Konsole können Sie für persistente Datastores einen Crawler hinzufügen, um den AWS Glue Data Catalog zu füllen. Sie können den Assistenten Add crawler (Crawler hinzufügen)

über die Liste der Tabellen oder die Liste der Crawler starten. Wählen Sie einen oder mehrere Datenspeicher aus, auf die Ihr Crawler zugreifen kann. Sie können auch einen Zeitplan erstellen, um zu bestimmen, wie häufig der Crawler ausgeführt wird. Für Datenstreams können Sie die Tabellendefinition manuell erstellen und Streameigenschaften definieren.

Optional können Sie einen benutzerdefinierten Classifier angeben, die das Schema Ihrer Daten ableitet. Sie können benutzerdefinierte Klassifizierer erstellen, indem Sie ein Grok-Muster verwenden. Jedoch AWS Glue bietet integrierte Klassifikatoren, die automatisch von Crawlern verwendet werden, wenn ein benutzerdefinierter Klassifikator Ihre Daten nicht erkennt. Wenn Sie einen Crawler definieren, müssen Sie keinen Classifier auswählen. Weitere Informationen zu Klassifikatoren finden Sie unter AWS Glue, finden Sie unter Klassifikatoren definieren und verwalten.

Für das Crawling bestimmter Arten von Datenspeichern ist eine Internetverbindung erforderlich, die Authentifizierungs- und Standortdaten bereitstellt. Bei Bedarf können Sie eine Verbindung herstellen, die diese erforderlichen Informationen in der AWS Glue console.

Der Crawler liest Ihren Datenspeicher und erstellt Datendefinitionen sowie benannte Tabellen im AWS Glue Data Catalog. Diese Tabellen werden in einer Datenbank Ihrer Wahl abgelegt. Sie können auch den Data Catalog mit manuell erstellten Tabellen füllen. Mit dieser Methode geben Sie das Schema und andere Metadaten an, um Tabellendefinitionen im Data Catalog zu erstellen. Da diese Methode ein wenig mühselig und fehleranfällig ist, ist es oft besser, die Tabellendefinitionen von einem Crawler erstellen zu lassen.

Weitere Hinweise zum Auffüllen der AWS Glue Data Catalog mit Tabellendefinitionen finden Sie unterErstellen von Tabellen.

2. Definieren Sie einen Auftrag, der die Transformation von Daten von der Quelle bis zum Ziel beschreibt.

   Im Allgemeinen müssen Sie zum Erstellen eines Auftrags die folgenden Optionen auswählen:

   - Wählen Sie eine Tabelle aus AWS Glue Data Catalog , die als Quelle für den Job dienen soll. Ihr Auftrag verwendet diese Tabellendefinition für den Zugriff auf Ihre Datenquelle und zum Interpretieren des Formats Ihrer Daten.
   - Wählen Sie eine Tabelle oder einen Speicherort aus AWS Glue Data Catalog , der das Ziel des Jobs sein soll. Ihr Auftrag verwendet diese Informationen für den Zugriff auf Ihren Datenspeicher.

- Erzähl AWS Glue um ein Skript zu generieren, um Ihre Quelle in ein Ziel umzuwandeln. AWS Glue generiert den Code zum Aufrufen integrierter Transformationen zur Konvertierung von Daten aus dem Quellschema in das Zielschemaformat. Diese Transformationen führen Vorgänge wie das Kopieren von Daten, Benennen von Spalten und Filtern von Daten aus, um Daten nach Bedarf zu transformieren. Sie können dieses Skript in der AWS Glue console.

Weitere Informationen zur Definition von Jobs finden Sie in AWS Glue, finden Sie unter Erstellung visueller ETL-Jobs.

3. Führen Sie Ihren Auftrag aus, um die Daten zu transformieren.

   Sie können Ihren Auftrag nach Bedarf ausführen oder basierend auf einem der folgenden Auslösertypen starten:

   - Ein Auslöser, der auf einem Cron-Zeitplan basiert.
   - Ein ereignisbasierter Trigger, z. B. kann der erfolgreiche Abschluss eines anderen Jobs einen auslösen AWS Glue Job.
   - Ein Auslöser zum Starten eines Auftrags nach Bedarf.

   Weitere Informationen zu Triggern finden Sie in AWS Glue, finden Sie unter Starten von Aufträgen und Crawlern über Auslöser.

4. Überwachen Sie Ihre geplanten Crawler und ausgelösten Aufträge.

   Verwenden Sie den AWS Glue Konsole, um Folgendes anzuzeigen:

   - Details und Fehler zu ausgeführten Aufträgen
   - Details und Fehler zu ausgeführten Crawlern
   - Irgendwelche Benachrichtigungen über AWS Glue Aktivitäten

   Weitere Informationen zur Überwachung Ihrer Crawler und Jobs finden Sie unter AWS Glue, finden Sie unter Überwachung AWS Glue.

Einrichten von IAM-Berechtigungen für AWS Glue

Die Anweisungen in diesem Thema helfen Ihnen beim schnellen Einrichten von AWS Identity and Access Management (IAM-) Berechtigungen für. AWS Glue Sie werden folgende Aufgaben erledigen:

- Gewähren Sie Ihren IAM-Identitäten Zugriff auf Ressourcen. AWS Glue

- Erstellen Sie eine Servicerolle für die Ausführung von Jobs, den Zugriff auf Daten und die Ausführung von AWS Glue Datenqualitätsaufgaben.

Ausführliche Anweisungen, mit denen Sie die IAM-Berechtigungen anpassen können AWS Glue, finden Sie unterKonfiguration von IAM-Berechtigungen für AWS Glue.

So richten Sie IAM-Berechtigungen für AWS Glue in der AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie Getting started (Erste Schritte).
3. Wählen Sie unter Konto vorbereiten für AWS Glue die Option IAM-Berechtigungen einrichten aus.
4. Wählen Sie die IAM-Identitäten (Rollen oder Benutzer) aus, denen Sie Berechtigungen erteilen AWS Glue möchten. AWS Glue hängt die `AWSGlueConsoleFullAccess` verwaltete Richtlinie an diese Identitäten an. Sie können diesen Schritt überspringen, wenn Sie diese Berechtigungen manuell festlegen möchten oder nur eine Standardeinstellung für die Servicerolle festlegen möchten.
5. Wählen Sie Weiter aus.
6. Wählen Sie die Amazon-S3-Zugriffsebene, die Ihre Rollen und Benutzer benötigen. Die Optionen, die Sie in diesem Schritt auswählen, werden auf alle von Ihnen ausgewählten Identitäten angewendet.

    a. Wählen Sie unter S3-Standorte auswählen die Amazon-S3-Standorte aus, auf die Sie Zugriff gewähren möchten.

    b. Wählen Sie als Nächstes aus, ob Ihre Identitäten nur Lesezugriff (empfohlen) oder Lese- und Schreibzugriff auf die zuvor ausgewählten Speicherorte haben sollen. AWS Glue fügt Ihren Identitäten basierend auf der von Ihnen ausgewählten Kombination aus Speicherorten und Lese- oder Schreibberechtigungen Berechtigungsrichtlinien hinzu.

    In der folgenden Tabelle sind die Berechtigungen aufgeführt, die AWS Glue für den Zugriff auf Amazon S3 gelten.

Wenn Sie auswählen ...	AWS Glue hängt an...
Keine Änderung	Keine Berechtigungen. AWS Glue wird keine Änderungen an den Berechtig ungen Ihrer Identität vornehmen.

Wenn Sie auswählen ...	AWS Glue hängt an...
Zugriff auf bestimmte Amazon-S3-Standorte gewähren (schreibgeschützt)	Die vom Kunden verwaltete Richtlinie gewährt Zugriff auf bestimmte Amazon S3 S3-Standorte mit Leseberechtigungen. `AWSGlueConsole-S3-read-only-policy` JSON **{** **"Version": "2012-10-17",** **"Statement": [** **{** **"Effect": "Allow",** **"Action": [** **"s3:GetObject"** **],** **"Resource": [** **"arn:aws:s3:::amzn-s3-demo-bucket1/*",** **"arn:aws:s3:::amzn-s3-demo-bucket2/*",** **"arn:aws:s3:::amzn-s3-demo-bucket3",** **"arn:aws:s3:::amzn-s3-demo-bucket"** **],** **"Condition": {** **"StringEquals": {** **"aws:ResourceAccount":"0000000000"** **}**

Wenn Sie auswählen ...	AWS Glue hängt an...

	`}`
	`}`
	`]`
	`}`

Wenn Sie auswählen ...	AWS Glue hängt an...
Zugriff auf bestimmte Amazon-S3-Standorte gewähren (Lese- und Schreibzugriff)	Das `AWSGlueConsole-S3-read-and-write-policy` gewährt Zugriff auf bestimmte Amazon S3 S3-Standorte mit Lese- und Schreibberechtigungen. JSON **{** **"Version": "2012-10-17",** **"Statement": [** **{** **"Effect": "Allow",** **"Action": [** **"s3:GetObject",** **"s3:PutObject"** **],** **"Resource": [** **"arn:aws:s3:::aes-siem-0000 0000000-log/*",** **"arn:aws:s3:::aes-siem-0000 0000000-snapshot/*",** **"arn:aws:s3:::aes-siem-0000 0000000-log",** **"arn:aws:s3:::aes-siem-0000 0000000-snapshot"** **],** **"Condition": {** **"StringEquals": {** **"aws:ResourceAccount":"00000000000"**

Wenn Sie auswählen ...	AWS Glue hängt an...
	} } }] }

7. Wählen Sie Weiter aus.
8. Wählen Sie eine AWS Glue Standard-Servicerolle für Ihr Konto. Eine Servicerolle ist eine IAM-Rolle, die AWS Glue verwendet wird, um in Ihrem Namen auf Ressourcen in anderen AWS Diensten zuzugreifen. Weitere Informationen finden Sie unter Servicerollen für AWS Glue.

   - Wenn Sie sich für die AWS Glue Standard-Servicerolle entscheiden, AWS Glue wird in Ihrem AWS-Konto Namen eine neue IAM-Rolle `AWSGlueServiceRole` mit den folgenden verwalteten Richtlinien erstellt. Wenn Ihr Konto bereits eine IAM-Rolle mit dem Namen hat`AWSGlueServiceRole`, werden diese Richtlinien AWS Glue an die bestehende Rolle angehängt.
     - AWSGlueServiceRole— Diese verwaltete Richtlinie ist erforderlich, AWS Glue damit Sie in Ihrem Namen auf Ressourcen zugreifen und diese verwalten können. Sie AWS Glue ermöglicht das Erstellen, Aktualisieren und Löschen verschiedener Ressourcen wie AWS Glue Jobs, Crawler und Verbindungen. Diese Richtlinie gewährt auch Berechtigungen für den Zugriff auf Amazon CloudWatch Protokolle AWS Glue zu Protokollierungszwecken. Für den Einstieg empfehlen wir, diese Richtlinie zu verwenden, um sich mit der Verwendung vertraut zu machen AWS Glue. Wenn Sie sich damit vertraut gemacht haben AWS Glue, können Sie Richtlinien erstellen, mit denen Sie den Zugriff auf Ressourcen nach Bedarf anpassen können.
     - AWSGlueConsoleFullAccess— Diese verwaltete Richtlinie gewährt vollen Zugriff auf den AWS Glue Service über die AWS Management Console. Diese Richtlinie gewährt Berechtigungen zur Ausführung aller darin enthaltenen AWS Glue Vorgänge und ermöglicht es Ihnen, jede AWS Glue Ressource nach Bedarf zu erstellen, zu ändern und zu löschen. Es ist jedoch wichtig zu beachten, dass diese Richtlinie keine Berechtigungen für den Zugriff auf die zugrunde liegenden Datenspeicher oder andere AWS Dienste gewährt, die möglicherweise am ETL-Prozess beteiligt sind. Aufgrund des breiten Umfangs der durch die `AWSGlueConsoleFullAccess` Richtlinie gewährten Berechtigungen sollte sie mit Vorsicht und unter Beachtung des Prinzips der geringsten Rechte vergeben werden. Es wird generell

empfohlen, nach Möglichkeit detailliertere Richtlinien zu erstellen und zu verwenden, die auf bestimmte Anwendungsfälle und Anforderungen zugeschnitten sind.

- AWSGlueConsole-S3- read-only-policy — Diese Richtlinie AWS Glue ermöglicht das Lesen von Daten aus bestimmten Amazon S3-Buckets, gewährt jedoch keine Berechtigungen zum Schreiben oder Ändern von Daten in Amazon S3 oder

  AWSGlueConsole-S3- read-and-write — Diese Richtlinie ermöglicht AWS Glue das Lesen und Schreiben von Daten in bestimmte Amazon S3 S3-Buckets als Teil des ETL-Prozesses.

- Wenn Sie eine bestehende IAM-Rolle auswählen, AWS Glue legt diese Rolle als Standard fest, fügt ihr jedoch keine `AWSGlueServiceRole` Berechtigungen hinzu. Stellen Sie sicher, dass Sie die Rolle so konfiguriert haben, dass sie als Servicerolle für AWS Glue verwendet werden soll. Weitere Informationen erhalten Sie unter Schritt 1: Erstellen Sie eine IAM-Richtlinie für den AWS Glue-Service und Schritt 2: Erstellen einer IAM-Rolle für AWS Glue.

9. Wählen Sie Weiter aus.
10. Überprüfen Sie abschließend die von Ihnen ausgewählten Berechtigungen und wählen Sie dann Änderungen übernehmen aus. Wenn Sie die Änderungen übernehmen, werden den ausgewählten Identitäten IAM-Berechtigungen AWS Glue hinzugefügt. Sie können die neuen Berechtigungen in der IAM-Konsole unter anzeigen oder ändern. https://console.aws.amazon.com/iam/

Sie haben jetzt die Einrichtung der IAM-Mindestberechtigungen für abgeschlossen. AWS Glue In einer Produktionsumgebung empfehlen wir Ihnen, sich mit den AWS Ressourcen für Ihren Anwendungsfall vertraut Identitäts- und Zugriffsmanagement für AWS Glue zu machen Sicherheit in AWS Glue und Ihnen zu helfen, diese zu sichern.

Nächste Schritte

Nachdem Sie nun die IAM-Berechtigungen eingerichtet haben, können Sie die folgenden Themen erkunden, um mit der Verwendung von AWS Glue zu beginnen:

- Erste Schritte mit AWS Glue in AWS Skill Builder
- Erste Schritte mit dem AWS Glue Data Catalog

Einrichtung für AWS Glue Studio

Erledigen Sie die Aufgaben in diesem Abschnitt, wenn Sie AWS Glue für das visuelle ETL zum ersten Mal:

Themen

- Erforderliche IAM-Berechtigungen für den AWS Glue Studio-Benutzer überprüfen
- Erforderliche IAM-Berechtigungen für ETL-Aufträge überprüfen
- Einstellen von IAM-Berechtigungen für AWS Glue Studio
- Konfigurieren Sie eine VPC für Ihren ETL-Auftrag

Erforderliche IAM-Berechtigungen für den AWS Glue Studio-Benutzer überprüfen

Für die Verwendung AWS Glue Studio muss der Benutzer Zugriff auf verschiedene AWS Ressourcen haben. Der Benutzer muss Amazon-S3-Buckets, IAM-Richtlinien und -Rollen sowie AWS Glue Data Catalog-Objekte anzeigen und auswählen können.

AWS Glue-Service-Berechtigungen

AWS Glue Studio verwendet die Aktionen und Ressourcen des AWS Glue-Services. Ihr Benutzer benötigt Berechtigungen für diese Aktionen und Ressourcen, um AWS Glue Studio sinnvoll verwenden zu können. Sie können für den AWS Glue Studio-Benutzer die verwaltete Richtlinie `AWSGlueConsoleFullAccess` aktivieren oder eine benutzerdefinierte Richtlinie mit weniger Berechtigungen erstellen.

> **Important**
>
> Im Sinne der Sicherheit hat es sich bewährt, den Zugriff auf Amazon-S3-Bucket- und Amazon CloudWatch -Protokoll-Gruppen durch strengere Richtlinien einzuschränken. Eine Amazon-S3-Beispielrichtlinie finden Sie unter Schreiben von IAM-Richtlinien: So gewähren Sie Zugriff auf einen Amazon-S3-Bucket.

Erstellen benutzerdefinierter IAM-Richtlinien für AWS Glue Studio

Sie können eine benutzerdefinierte Richtlinie mit weniger Berechtigungen für AWS Glue Studio erstellen. Die Richtlinie kann Berechtigungen für eine Teilmenge von Objekten oder Aktionen erteilen. Verwenden Sie die folgenden Informationen, wenn Sie eine benutzerdefinierte Richtlinie erstellen.

Um das verwenden zu können AWS Glue Studio APIs, nehmen Sie `glue:UseGlueStudio` in der Aktionsrichtlinie Ihre IAM-Berechtigungen auf. Die Verwendung von `glue:UseGlueStudio` ermöglicht Ihnen den Zugriff auf alle AWS Glue Studio-Aktionen, auch wenn im Laufe der Zeit weitere Aktionen zur API hinzugefügt werden.

Weitere Informationen zu Aktionen, die von definiert sind AWS Glue, finden Sie unter Aktionen definiert von AWS Glue.

Aktionen zur Datenaufbereitung und -erstellung

- SendRecipeAction
- GetRecipeAction

DAG-Aktionen (Ausgerichtetes azyklisches Diagramm)

- CreateDag
- UpdateDag
- GetDag
- DeleteDag

Auftragsaktionen

- SaveJob
- GetJob
- CreateJob
- DeleteJob
- GetJobs
- UpdateJob

Aktionen zur Auftragsausführung

- StartJobRun
- GetJobRuns
- BatchStopJobRun
- GetJobRun

- QueryJobRuns
- QueryJobs
- QueryJobRunsAggregated

Schema-Aktionen

- GetSchema
- GetInferredSchema

Datenbank-Aktionen

- GetDatabases

Plan-Aktionen

- GetPlan

Tabellen-Aktionen

- SearchTables
- GetTables
- GetTable

Verbindungs-Aktionen

- CreateConnection
- DeleteConnection
- UpdateConnection
- GetConnections
- GetConnection

Zuordnungs-Aktionen

- GetMapping

S3-Proxy-Aktionen

- ListBuckets
- ListObjectsV2
- GetBucketLocation

Sicherheitskonfigurations-Aktionen

- GetSecurityConfigurations

Skript-Aktionen

- CreateScript (unterscheidet sich von der gleichnamigen API inAWS Glue)

Zugriff auf AWS Glue Studio APIs

Um auf AWS Glue Studio zuzugreifen, fügen Sie `glue:UseGlueStudio` in der Liste der Aktionsrichtlinien in den IAM-Berechtigungen hinzu.

Im Beispiel unten sind sie in der Aktionsrichtlinie enthalten, aber AWS Glue Studio APIs sie werden nicht einzeln identifiziert. `glue:UseGlueStudio` Das liegt daran`glue:UseGlueStudio`, dass Sie beim Einbeziehen automatisch Zugriff auf die internen Daten erhalten, APIs ohne dass Sie die Person AWS Glue Studio APIs in den IAM-Berechtigungen angeben müssen.

In diesem Beispiel sind die zusätzlich aufgelisteten Aktionsrichtlinien (z. B.`glue:SearchTables`) nicht vorhanden AWS Glue Studio APIs, sodass sie bei Bedarf in die IAM-Berechtigungen aufgenommen werden müssen. Sie können auch Amazon-S3-Proxy-Aktionen einschließen, um die Ebene des zu gewährenden Amazon-S3-Zugriffs festzulegen. Die folgende Beispielrichtlinie ermöglicht den Zugriff auf das Öffnen AWS Glue Studio und save/run Erstellen eines visuellen Auftrags, sofern die ausgewählte IAM-Rolle über ausreichenden Zugriff verfügt.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
```

```
            "Effect": "Allow",
            "Action": [
                "glue:UseGlueStudio",
                "iam:ListRoles",
                "iam:ListUsers",
                "iam:ListGroups",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "glue:SearchTables",
                "glue:GetConnections",
                "glue:GetJobs",
                "glue:GetTables",
                "glue:BatchStopJobRun",
                "glue:GetSecurityConfigurations",
                "glue:DeleteJob",
                "glue:GetDatabases",
                "glue:CreateConnection",
                "glue:GetSchema",
                "glue:GetTable",
                "glue:GetMapping",
                "glue:CreateJob",
                "glue:DeleteConnection",
                "glue:CreateScript",
                "glue:UpdateConnection",
                "glue:GetConnection",
                "glue:StartJobRun",
                "glue:GetJobRun",
                "glue:UpdateJob",
                "glue:GetPlan",
                "glue:GetJobRuns",
                "glue:GetTags",
                "glue:GetJob",
                "glue:QueryJobRuns",
                "glue:QueryJobs",
                "glue:QueryJobRunsAggregated",
                "glue:SendRecipeAction",
                "glue:GetRecipeAction"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "iam:PassRole"
```

```
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Berechtigungen für Notebook und Datenvorschau

Mit Datenvorschauen und Notizbüchern können Sie in jeder Phase Ihres Auftrags (Lesen, Transformieren, Schreiben) ein Beispiel Ihrer Daten anzeigen, ohne den Auftrag ausführen zu müssen. Sie geben eine AWS Identity and Access Management (IAM-) Rolle an, die beim Zugriff auf die Daten verwendet werden AWS Glue Studio soll. IAM-Rollen sollen annehmbar sein und haben keine langfristigen Standard-Anmeldeinformationen wie ein Kennwort oder damit verbundene Zugriffsschlüssel. Wenn AWS Glue Studio die Rolle übernimmt, stellt IAM stattdessen temporäre Sicherheitsanmeldeinformationen bereit.

Um sicherzustellen, dass Datenvorschauen und Notebook-Befehle ordnungsgemäß funktionieren, verwenden Sie eine Rolle mit einem Namen, der mit der Zeichenfolge `AWSGlueServiceRole` beginnt. Wenn Sie einen anderen Namen für Ihre Rolle verwenden möchten, müssen Sie die `iam:passrole`-Berechtigung hinzufügen und eine Richtlinie für die Rolle in IAM konfigurieren. Weitere Informationen finden Sie unter Erstellen Sie eine IAM-Richtlinie für Rollen, die nicht "AWSGlueServiceRole*“ heißen.

> **Warning**
>
> Wenn eine Rolle die `iam:passrole`-Berechtigung für ein Notebook gewährt und Sie eine Rollenverkettung implementieren, könnte ein Benutzer unbeabsichtigt Zugriff auf das Notebook erlangen. Derzeit ist kein Auditing implementiert, mit dem Sie überwachen können, welchen Benutzern Zugriff auf das Notebook gewährt wurde.

Wenn Sie einer IAM-Identität die Möglichkeit verweigern möchten, Datenvorschau-Sitzungen zu erstellen, lesen Sie das folgende Beispiel the section called “Einer Identität die Möglichkeit verweigern, Datenvorschau-Sitzungen zu erstellen”.

Amazon CloudWatch -Berechtigungen

Sie können Ihre AWS Glue Studio Jobs mithilfe von Metriken überwachen Amazon CloudWatch, die Rohdaten sammeln und AWS Glue zu lesbaren near-real-time Metriken verarbeiten. Standardmäßig werden AWS Glue Metrikdaten CloudWatch automatisch an diese gesendet. Weitere Informationen finden Sie unter Was ist Amazon CloudWatch? im CloudWatch Amazon-Benutzerhandbuch und AWS GlueMetrics im AWS Glue Entwicklerhandbuch.

Um auf CloudWatch Dashboards zugreifen zu können, AWS Glue Studio benötigt der zugreifende Benutzer eine der folgenden Voraussetzungen:

- Die Richtlinie `AdministratorAccess`
- Die Richtlinie `CloudWatchFullAccess`
- Eine benutzerdefinierte Richtlinie mit einem oder mehreren dieser spezifischen Berechtigungen:
  - `cloudwatch:GetDashboard` und `cloudwatch:ListDashboards` zum Anzeigen von Dashboards
  - `cloudwatch:PutDashboard` zum Erstellen bzw. Ändern von Dashboards
  - `cloudwatch:DeleteDashboards` zum Löschen von Dashboards

Weitere Informationen zum Ändern von Berechtigungen für einen IAM-Benutzer mithilfe von Richtlinien finden Sie unter Ändern von Berechtigungen für einen IAM-Benutzer im IAM-Benutzerhandbuch.

Erforderliche IAM-Berechtigungen für ETL-Aufträge überprüfen

Wenn Sie einen Job erstellen mit AWS Glue Studio, nimmt der Job die Berechtigungen der IAM-Rolle an, die Sie bei der Erstellung angeben. Diese IAM-Rolle muss berechtigt sein, Daten aus Ihrer Datenquelle zu extrahieren, Daten in Ihr Ziel zu schreiben und auf Ressourcen zuzugreifen AWS Glue .

Der Name der Rolle, die Sie für den Job erstellen, muss mit der Zeichenfolge `AWSGlueServiceRole` beginnen, damit er korrekt verwendet werden kann AWS Glue Studio. Sie könnten beispielsweise Ihre Rolle benennen`AWSGlueServiceRole-FlightDataJob`.

Datenquellen- und Datenzielberechtigungen

Importieren in &S3; AWS Glue Studio Der Job muss Zugriff auf Amazon S3 für alle Quellen, Ziele, Skripten und temporären Verzeichnisse haben, die Sie in Ihrem Job verwenden. Sie können eine Richtlinie erstellen, die einen differenzierten Zugriff auf bestimmte Amazon-S3-Ressourcen ermöglicht.

- Datenquellen erfordern die Berechtigungen `s3:ListBucket` und `s3:GetObject`.
- Für Datenziele sind die Berechtigungen `s3:ListBucket`, `s3:PutObject` und `s3:DeleteObject` erforderlich.

> **Note**
>
> Ihre IAM-Richtlinie muss die spezifischen Buckets berücksichtigen, die `s3:GetObject` für das Hosten AWS Glue von Transformationen verwendet werden.
> Die folgenden Buckets gehören dem AWS Dienstkonto und sind weltweit lesbar. Diese Buckets dienen als Repository für den Quellcode einer Teilmenge von Transformationen, auf die über den visuellen Editor zugegriffen werden kann. AWS Glue Studio Die Berechtigungen für den Bucket sind so eingerichtet, dass jede andere API-Aktion für den Bucket verweigert wird. Jeder kann die Skripte lesen, die wir für die Transformationen bereitstellen, aber niemand außerhalb unseres Serviceteams kann etwas in sie „einbauen“. Wenn Ihr AWS Glue Job ausgeführt wird, wird diese Datei als lokaler Import abgerufen, sodass die Datei in den lokalen Container heruntergeladen wird. Danach findet keine weitere Kommunikation mit diesem Konto statt.

Region: Bucket-Name

- af-south-1: aws-glue-studio-transforms -762339736633- -1 prod-af-south
- ap-east-1: aws-glue-studio-transforms -125979764932- -1 prod-ap-east
- ap-northeast-2: -673535381443- -2 aws-glue-studio-transforms prod-ap-northeast
- ap-northeast-3: -149976050262- -3 aws-glue-studio-transforms prod-ap-northeast
- ap-south-1: aws-glue-studio-transforms -584702181950- -1 prod-ap-south
- ap-south-2: aws-glue-studio-transforms -380279651983- -2 prod-ap-south
- ap-southeast-1: -737106620487- -1 aws-glue-studio-transforms prod-ap-southeast
- ap-southeast-2: -234881715811- -2 aws-glue-studio-transforms prod-ap-southeast

- ap-southeast-3: -151265630221- -3 aws-glue-studio-transforms prod-ap-southeast
- ap-southeast-4: -052235663858- -4 aws-glue-studio-transforms prod-ap-southeast
- ca-central-1: aws-glue-studio-transforms -622716468547- -1 prod-ca-central
- ca-west-1: aws-glue-studio-transforms -915795495192- -1 prod-ca-west
- eu-central-1: aws-glue-studio-transforms -560373232017- -1 prod-eu-central
- eu-central-2: aws-glue-studio-transforms -907358657121- -2 prod-eu-central
- eu-north-1: -312557305497 aws-glue-studio-transforms - -1 prod-eu-north
- eu-south-1: aws-glue-studio-transforms -939684186351- -1 prod-eu-south
- eu-south-2: aws-glue-studio-transforms -239737454084- -2 prod-eu-south
- eu-west-1: aws-glue-studio-transforms -244479516193- -1 prod-eu-west
- eu-west-2: aws-glue-studio-transforms -804222392271- -2 prod-eu-west
- eu-west-3: aws-glue-studio-transforms -371299348807- -3 prod-eu-west
- il-central-1: aws-glue-studio-transforms -806964611811- -1 prod-il-central
- me-central-1: aws-glue-studio-transforms -733304270342- -1 prod-me-central
- me-south-1: -112120182341 aws-glue-studio-transforms - -1 prod-me-south
- sa-east-1: aws-glue-studio-transforms -881619130292- -1 prod-sa-east
- us-east-1: aws-glue-studio-transforms -510798373988- -1 prod-us-east
- us-east-2: aws-glue-studio-transforms -251189692203- -2 prod-us-east
- us-west-1: aws-glue-studio-transforms -593230150239- -1 prod-us-west
- us-west-2: aws-glue-studio-transforms -818035625594- -2 prod-us-west
- ap-northeast-1: -200493242866- -1 aws-glue-studio-transforms prod-ap-northeast
- aws-glue-studio-transformscn-north-1: -071033555442- -1 prod-cn-north
- cn-northwest-1: aws-glue-studio-transforms -070947029561- -1 prod-cn-northwest
- us-gov-west-1: aws-glue-studio-transforms -227493901923- -1-2604 prod-us-gov-west

Wenn Sie Amazon Redshift als Datenquelle wählen, können Sie eine Rolle für Clusterberechtigungen angeben. Jobs, die für einen Amazon Redshift Cluster ausgeführt werden, geben Befehle aus, die mithilfe temporärer Anmeldeinformationen auf Amazon S3 für temporären Speicher zugreifen. Wenn Ihr Auftrag länger als eine Stunde ausgeführt wird, laufen diese Anmeldeinformationen ab, wodurch der Auftrag fehlschlägt. Um dieses Problem zu vermeiden, können Sie dem Amazon Redshift -Cluster selbst eine Rolle zuweisen, die ihm die erforderlichen Berechtigungen für Aufträge mit temporären

Anmeldeinformationen erteilt. Weitere Informationen finden Sie unter Verschieben von Daten von und nach Amazon Redshift im AWS Glue -Entwicklerhandbuch.

Wenn der Auftrag andere Datenquellen oder Ziele als Amazon S3 verwendet, müssen Sie der IAM-Rolle, die der Auftrag verwendet, die erforderlichen Berechtigungen erteilen, damit sie auf diese Datenquellen und Ziele zuzugreifen kann. Weitere Informationen finden Sie unter Einrichten der Umgebung, um auf Datenspeicher zuzugreifen im AWS Glue -Entwicklerhandbuch.

Wenn Sie Konnektoren und Verbindungen für Ihren Datenspeicher verwenden, benötigen Sie zusätzliche Berechtigungen, wie unter the section called “Erforderliche Berechtigungen zur Verwendung von Konnektoren” beschrieben.

Erforderliche Berechtigungen zum Löschen von Aufträgen

In AWS Glue Studio Sie können in der Konsole mehrere Jobs zum Löschen auswählen. Um diese Aktion ausführen zu können, müssen Sie über die Berechtigung `glue:BatchDeleteJob` verfügen. Das unterscheidet sich von AWS Glue Konsole, für die die `glue:DeleteJob` Erlaubnis zum Löschen von Jobs erforderlich ist.

AWS Key Management Service Berechtigungen

Wenn Sie vorhaben, auf Amazon S3 S3-Quellen und -Ziele zuzugreifen, die serverseitige Verschlüsselung mit AWS Key Management Service (AWS KMS) verwenden, fügen Sie eine Richtlinie an AWS Glue Studio Die vom Job verwendete Rolle, die es dem Job ermöglicht, die Daten zu entschlüsseln. Die Auftragsrolle benötigt die Berechtigungen `kms:ReEncrypt`, `kms:GenerateDataKey` und `kms:DescribeKey`. Darüber hinaus benötigt die Jobrolle die `kms:Decrypt` Erlaubnis, ein Amazon S3 S3-Objekt hoch- oder herunterzuladen, das mit einem AWS KMS Kundenhauptschlüssel (CMK) verschlüsselt ist.

Für die Nutzung AWS KMS CMKs fallen zusätzliche Gebühren an. Weitere Informationen finden Sie im AWS Key Management Service Entwicklerhandbuch unter AWS Key Management Service Konzepte — Kundenhauptschlüssel (CMKs) und AWS Key Management Service Preise.

Erforderliche Berechtigungen zur Verwendung von Konnektoren

Wenn Sie eine verwenden AWS Glue Benutzerdefinierter Connector und Verbindung für den Zugriff auf einen Datenspeicher, die Rolle, mit der der AWS Glue Für den ETL-Job sind zusätzliche Berechtigungen erforderlich:

- Die von AWS verwaltete Richtlinie `AmazonEC2ContainerRegistryReadOnly` für den Zugriff auf Connectors, bei denen gekauft wurde AWS Marketplace.

- Die Berechtigungen `glue:GetJob` und `glue:GetJobs`.
- AWS Secrets Manager Berechtigungen für den Zugriff auf geheime Daten, die bei Verbindungen verwendet werden. Informationen zu IAM-Richtlinien erhalten Sie unter Beispiel: Berechtigung zum Abrufen von Secret-Werten.

Wenn Ihre AWS Glue Der ETL-Job wird in einer VPC ausgeführt, auf der Amazon VPC ausgeführt wird. Anschließend muss die VPC wie unter beschrieben konfiguriert werden. the section called “Konfigurieren Sie eine VPC für Ihren ETL-Auftrag”

Einstellen von IAM-Berechtigungen für AWS Glue Studio

Sie können als AWS -Administrator die Rollen erstellen und Richtlinien an Benutzer und Auftragsrollen zuweisen.

Sie können die AWSGlueConsoleFullAccess AWS verwaltete Richtlinie verwenden, um die erforderlichen Berechtigungen für die Verwendung der AWS Glue Studio Konsole bereitzustellen.

Führen Sie zum Erstellen einer eigenen Richtlinie die Schritte unter Erstellen einer IAM-Richtlinie für den AWS Glue-Service im AWS Glue -Entwicklerhandbuch aus. Fügen Sie die zuvor in Erforderliche IAM-Berechtigungen für den AWS Glue Studio-Benutzer überprüfen beschriebenen IAM-Berechtigungen ein.

Themen

- Dem AWS Glue Studio-Benutzer Richtlinien zuweisen
- Erstellen Sie eine IAM-Richtlinie für Rollen, die nicht "AWSGlueServiceRole*“ heißen

Dem AWS Glue Studio-Benutzer Richtlinien zuweisen

Jeder AWS Benutzer, der sich an der AWS Glue Studio Konsole anmeldet, muss über Berechtigungen für den Zugriff auf bestimmte Ressourcen verfügen. Sie erteilen diese Berechtigungen, indem Sie dem Benutzer IAM-Richtlinien zuweisen.

So fügen Sie die AWSGlueConsoleFullAccess-verwaltete Richtlinie einem Benutzer hinzu

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien.

3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben dem AWSGlueConsoleFullAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).
6. Wiederholen Sie die vorherigen Schritte, um dem Benutzer nach Bedarf zusätzliche Richtlinien zuzuweisen.

Erstellen Sie eine IAM-Richtlinie für Rollen, die nicht "AWSGlueServiceRole*“ heißen

Konfigurieren einer IAM-Richtlinie für Rollen, die von AWS Glue Studio verwendet werden

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Fügen Sie eine neue IAM-Richtlinie hinzu. Sie können eine vorhandene Richtlinie hinzufügen oder eine neue eingebundene IAM-Richtlinie erstellen. Erstellen Sie eine IAM-Richtlinie wie folgt:
    1. Wählen Sie Policies aus und wählen Sie dann Create Policy aus. Wenn die Schaltfläche Get Started angezeigt wird, klicken Sie darauf und wählen Sie anschließend Create Policy.
    2. Klicken Sie neben Create Your Own Policy auf Select.
    3. Geben Sie im Feld Policy Name einen beliebigen Wert ein, den Sie später bequem verwenden können. Optional können Sie auch im Feld Description eine Beschreibung eingeben.
    4. Geben Sie im Feld Policy Document eine Richtlinienanweisung im folgendem Format ein und wählen Sie Create Policy aus:
3. Kopieren Sie die folgenden Blöcke und fügen Sie sie in die Richtlinie unter dem Array „Statement“ ein. *my-interactive-session-role-prefix* Ersetzen Sie sie durch das Präfix für alle gängigen Rollen, für AWS Glue die Berechtigungen zugewiesen werden sollen.

```
{
    "Action": [
        "iam:PassRole"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/my-interactive-session-role-prefix*",
    "Condition": {
```

```
        "StringLike": {
            "iam:PassedToService": [
                "glue.amazonaws.com "
            ]
        }
    }
}
```

Hier ist das vollständige Beispiel mit den Versions- und Anweisungs-Arrays, die in der Richtlinie enthalten sind

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Action": [
            "iam:PassRole"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:iam::*:role/my-interactive-session-role-
prefix*",
        "Condition": {
            "StringLike": {
                "iam:PassedToService": [
                    "glue.amazonaws.com "
                ]
            }
        }
    }
  ]
}
```

4. Wählen Sie Users aus, um die Richtlinie für einen Benutzer zu aktivieren.
5. Wählen Sie den -Benutzer aus, an den Sie die Richtlinie anfügen möchten.

Konfigurieren Sie eine VPC für Ihren ETL-Auftrag

Sie können Amazon Virtual Private Cloud (Amazon VPC) verwenden, um ein virtuelles Netzwerk in Ihrem eigenen logisch isolierten Bereich innerhalb der AWS Cloud sogenannten Virtual Private Cloud

(VPC) zu definieren. Sie können Ihre AWS -Ressourcen, z. B. Instances, in Ihrer VPC launchen. Eine VPC ist einem herkömmlichen Netzwerk in einem eigenen Rechenzentrum sehr ähnlich, bietet jedoch die Vorteile durch die Nutzung der skalierbaren Infrastruktur von AWS. Sie können Ihre VPC konfigurieren. Hierzu können Sie den IP-Adressbereich auswählen, Subnetze erstellen sowie Routing-Tabellen, Netzwerk-Gateways und Sicherheitseinstellungen konfigurieren. Sie können jetzt Instances in der VPC mit dem Internet verbinden. Sie können Ihre VPC mit Ihrem eigenen Unternehmensrechenzentrum verbinden und sie so zu AWS Cloud einer Erweiterung Ihres Rechenzentrums machen. Um die Ressourcen in den einzelnen Subnetzen zu schützen, können Sie mehrere Sicherheitsebenen verwenden, darunter Sicherheitsgruppen und Netzwerk-Zugriffskontrolllisten. Weitere Informationen finden Sie im Amazon-VPC-Benutzerhandbuch.

Sie können Ihre konfigurieren AWS Glue ETL-Jobs, die innerhalb einer VPC ausgeführt werden sollen, wenn Connectors verwendet werden. Sie müssen Ihre VPC bei Bedarf für Folgendes konfigurieren:

- Öffentlicher Netzwerkzugriff für Datenspeicher, die nicht vorhanden sind. AWS Alle Datenspeicher, auf die der Auftrag zugreift, müssen über das VPC-Subnetz verfügbar sein.
- Wenn Ihr Auftrag sowohl auf VPC-Ressourcen als auch auf das öffentliche Internet zugreifen muss, muss die VPC über ein NAT-Gateway (Network Address Translation) innerhalb der VPC verfügen.

  Weitere Informationen finden Sie unter Einrichten der Umgebung, um auf Datenspeicher zuzugreifen im AWS Glue -Entwicklerhandbuch.

Erste Schritte mit Notebooks in AWS Glue Studio

Wenn Sie ein Notebook über AWS Glue Studio starten, werden alle Konfigurationsschritte für Sie ausgeführt, damit Sie Ihre Daten untersuchen und nach wenigen Sekunden mit der Entwicklung Ihres Auftragsskripts beginnen können.

In den folgenden Abschnitten wird beschrieben, wie Sie eine Rolle erstellen und die entsprechenden Berechtigungen gewähren, um Notebooks in AWS Glue Studio für ETL-Aufträge zu verwenden.

Weitere Informationen zu den von AWS Glue definierten Aktionen finden Sie unter Von AWS Glue definierte Aktionen.

Themen

- Erteilen von Berechtigungen für die IAM-Rolle

Erteilen von Berechtigungen für die IAM-Rolle

Das Einrichten von AWS Glue Studio ist eine Voraussetzung für die Verwendung von Notebooks.

Um Notebooks in AWS Glue verwenden zu können, erfordert Ihre Rolle Folgendes:

- Eine Vertrauensstellung zu AWS Glue für die Aktion `sts:AssumeRole` bzw. `sts:TagSession`, falls Markierungen gewünscht sind.
- Eine IAM-Richtlinie, die alle Berechtigungen für Notizbücher und interaktive Sitzungen enthält. AWS Glue
- Eine IAM-Richtlinie für eine Rollenübergabe, da die Rolle sich selbst vom Notebook an interaktive Sitzungen weitergeben können muss.

Wenn Sie beispielsweise eine neue Rolle erstellen, können Sie der Rolle eine AWS verwaltete Standardrichtlinie `AWSGlueConsoleFullAccessRole` hinzufügen und dann eine neue Richtlinie für den Notebook-Betrieb und eine weitere für die PassRole IAM-Richtlinie hinzufügen.

Erforderliche Maßnahmen für eine Vertrauensstellung zu AWS Glue

Wenn Sie eine Notebook-Sitzung starten, müssen Sie `sts:AssumeRole` zur Vertrauensstellung der Rolle hinzufügen, die an das Notebook weitergegeben wird. Wenn Ihre Sitzung Tags enthält, müssen Sie auch die Aktion `sts:TagSession` weitergeben. Ohne diese Aktionen kann die Notebook-Sitzung nicht gestartet werden.

Zum Beispiel:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
```

```
}
```

Richtlinien, die IAM-Berechtigungen für Notebooks enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für Notebooks. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartNotebook",
                "glue:TerminateNotebook",
                "glue:GlueNotebookRefreshCredentials",
                "glue:DeregisterDataPreview",
                "glue:GetNotebookInstanceStatus",
                "glue:GlueNotebookAuthorize"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können die folgenden IAM-Richtlinien verwenden, um den Zugriff auf bestimmte Ressourcen zu gewähren:

- AwsGlueSessionUserRestrictedNotebookServiceRole: Bietet vollen Zugriff auf alle AWS Glue Ressourcen mit Ausnahme von Sitzungen. Ermöglicht Benutzern, nur die Notebook-Sitzungen zu erstellen und zu verwenden, die mit dem Benutzer verknüpft sind. Diese Richtlinie umfasst auch andere Berechtigungen, die für AWS Glue die Verwaltung von AWS Glue Ressourcen in anderen AWS Diensten erforderlich sind.
- AwsGlueSessionUserRestrictedNotebookPolicy: Stellt Berechtigungen bereit, die es Benutzern ermöglichen, nur die Notizbuchsitzungen zu erstellen und zu verwenden, die dem Benutzer

zugeordnet sind. Diese Richtlinie enthält auch explizite Berechtigungen zur Weitergabe einer eingeschränkten AWS Glue-Sitzungsrolle durch Benutzer.

IAM-Richtlinie zur Übergabe einer Rolle

Wenn Sie ein Notebook mit einer Rolle erstellen, wird diese Rolle dann an interaktive Sitzungen übergeben, sodass dieselbe Rolle an beiden Stellen verwendet werden kann. Daher muss die `iam:PassRole`-Berechtigung Teil der Richtlinie der Rolle sein.

Erstellen Sie anhand des folgenden Beispiels eine neue Richtlinie für Ihre Rolle. Ersetzen Sie die Kontonummer durch Ihre eigene und den Rollennamen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::090000000210:role/<role_name>"
        }
    ]
}
```

AWS Glue Nutzungsprofile einrichten

Einer der Hauptvorteile der Verwendung einer Cloud-Plattform ist ihre Flexibilität. Mit dieser einfachen Erstellung von Rechenressourcen besteht jedoch die Gefahr, dass die Cloud-Kosten in die Höhe schnellen, wenn sie nicht verwaltet werden und keine Schutzmaßnahmen ergriffen werden. Aus diesem Grund müssen Administratoren die Balance finden, hohe Infrastrukturkosten zu vermeiden und gleichzeitig den Benutzern ein reibungsloses Arbeiten zu ermöglichen.

Mithilfe AWS Glue von Nutzungsprofilen können Administratoren verschiedene Profile für verschiedene Benutzerklassen innerhalb des Kontos erstellen, z. B. für Entwickler, Tester und Produktteams. Jedes Profil besteht aus einem eindeutigen Satz von Parametern, die verschiedenen Benutzertypen zugewiesen werden können. Beispielsweise benötigen Entwickler möglicherweise

mehr Mitarbeiter und können eine höhere maximale Anzahl an Mitarbeitern haben, während Produktteams möglicherweise weniger Mitarbeiter und einen niedrigeren Timeout- oder Leerlauf-Timeout-Wert benötigen.

Beispiel für das Verhalten von Aufträgen und Auftragsausführungen

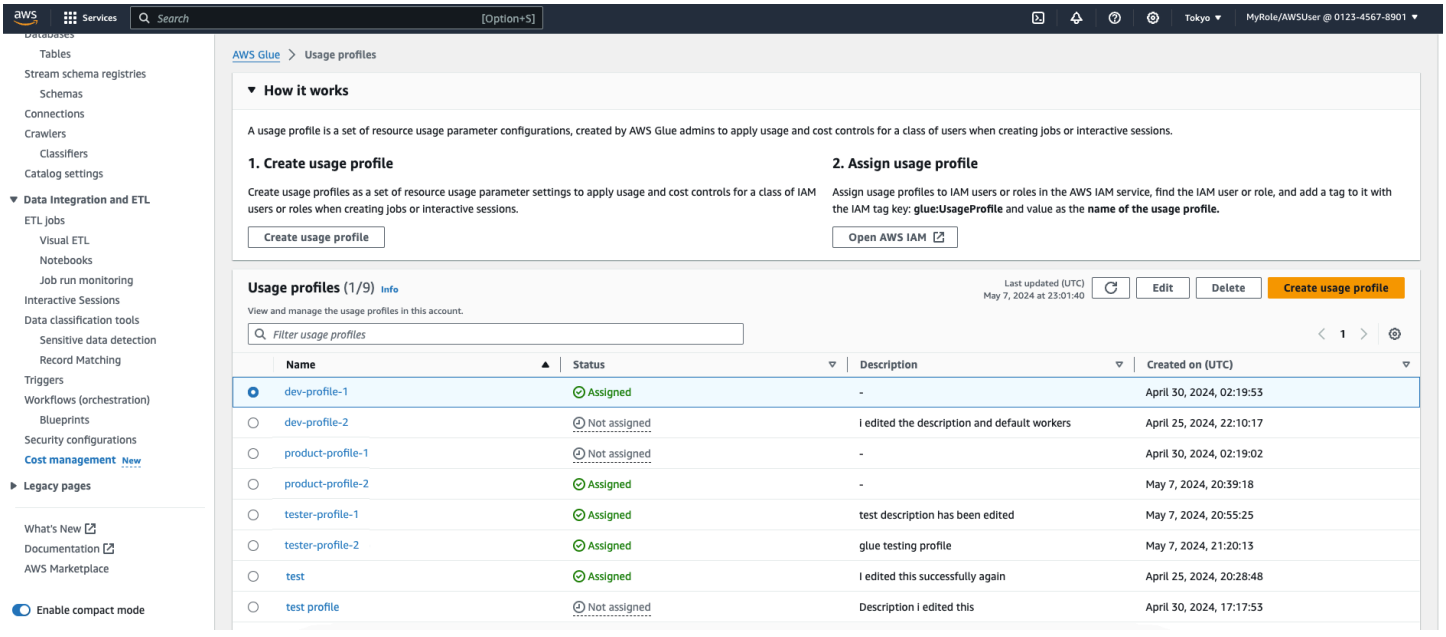
Angenommen, ein Job wird von Benutzer A mit Profil A erstellt. Der Job wird mit bestimmten Parameterwerten gespeichert. Benutzer B mit Profil B versucht, den Job auszuführen.

Wenn Benutzer A den Job verfasste und er keine bestimmte Anzahl von Mitarbeitern festlegte, wurde der im Profil von Benutzer A festgelegte Standard angewendet und zusammen mit den Jobdefinitionen gespeichert.

Wenn Benutzer B den Job ausführt, wird er mit den Werten ausgeführt, die für ihn gespeichert wurden. Wenn das eigene Profil von Benutzer B restriktiver ist und es nicht erlaubt ist, mit so vielen Mitarbeitern zu arbeiten, schlägt die Ausführung des Jobs fehl.

Nutzungsprofil als Ressource

Ein AWS Glue Nutzungsprofil ist eine Ressource, die durch einen Amazon-Ressourcennamen (ARN) identifiziert wird. Es gelten alle standardmäßigen IAM-Kontrollen (Identity and Access Management), einschließlich aktionsbasierter und ressourcenbasierter Autorisierung. Administratoren sollten die IAM-Richtlinie der Benutzer, die AWS Glue Ressourcen erstellen, aktualisieren und ihnen Zugriff auf die Nutzung der Profile gewähren.



Themen

- Nutzungsprofile erstellen und verwalten
- Nutzungsprofile und Jobs

Nutzungsprofile erstellen und verwalten

Ein AWS Glue Nutzungsprofil erstellen

Administratoren sollten Nutzungsprofile erstellen und diese dann den verschiedenen Benutzern zuweisen. Bei der Erstellung eines Nutzungsprofils geben Sie Standardwerte sowie einen Bereich zulässiger Werte für verschiedene Job- und Sitzungsparameter an. Sie müssen mindestens einen Parameter für Jobs oder interaktive Sitzungen konfigurieren. Sie können den Standardwert anpassen, der verwendet werden soll, wenn kein Parameterwert für den Job angegeben wird, and/or und eine Bereichsbeschränkung oder eine Reihe zulässiger Werte für die Überprüfung einrichten, falls ein Benutzer bei der Verwendung dieses Profils einen Parameterwert angibt.

Standardwerte sind eine bewährte Methode, die vom Administrator festgelegt wurde, um Jobautoren zu unterstützen. Wenn ein Benutzer einen neuen Job erstellt und keinen Timeout-Wert festlegt, gilt das Standard-Timeout des Nutzungsprofils. Wenn der Autor kein Profil hat, gelten die Standardwerte für den AWS Glue Dienst und werden in der Jobdefinition gespeichert. AWS Glue Erzwingt zur Laufzeit die im Profil festgelegten Grenzwerte (Mindest-, Maximal-, zulässige Anzahl an Mitarbeitern).

Sobald ein Parameter konfiguriert ist, sind alle anderen Parameter optional. Folgende Parameter können für Jobs oder interaktive Sitzungen angepasst werden:

- Anzahl der Mitarbeiter — Beschränken Sie die Anzahl der Mitarbeiter, um eine übermäßige Nutzung von Rechenressourcen zu vermeiden. Sie können einen Standard-, Mindest- und Höchstwert festlegen. Der Mindestwert ist 1.
- Mitarbeitertyp — schränken Sie die relevanten Mitarbeitertypen für Ihre Workloads ein. Sie können einen Standardtyp festlegen und Mitarbeitertypen für ein Benutzerprofil zulassen.
- Timeout — Definieren Sie die maximale Zeit, für die ein Job oder eine interaktive Sitzung ausgeführt werden kann und Ressourcen verbraucht, bevor sie beendet wird. Richten Sie Timeout-Werte ein, um lange laufende Jobs zu vermeiden.

  Sie können einen Standard-, Mindest- und Höchstwert in Minuten festlegen. Der Mindestwert ist 1 (Minute). Das AWS Glue Standard-Timeout beträgt zwar 2880 Minuten, Sie können jedoch einen beliebigen Standardwert im Nutzungsprofil festlegen.

Es hat sich bewährt, einen Wert für „Standard“ festzulegen. Dieser Wert wird für die Erstellung des Jobs oder der Sitzung verwendet, wenn vom Benutzer kein Wert festgelegt wurde.

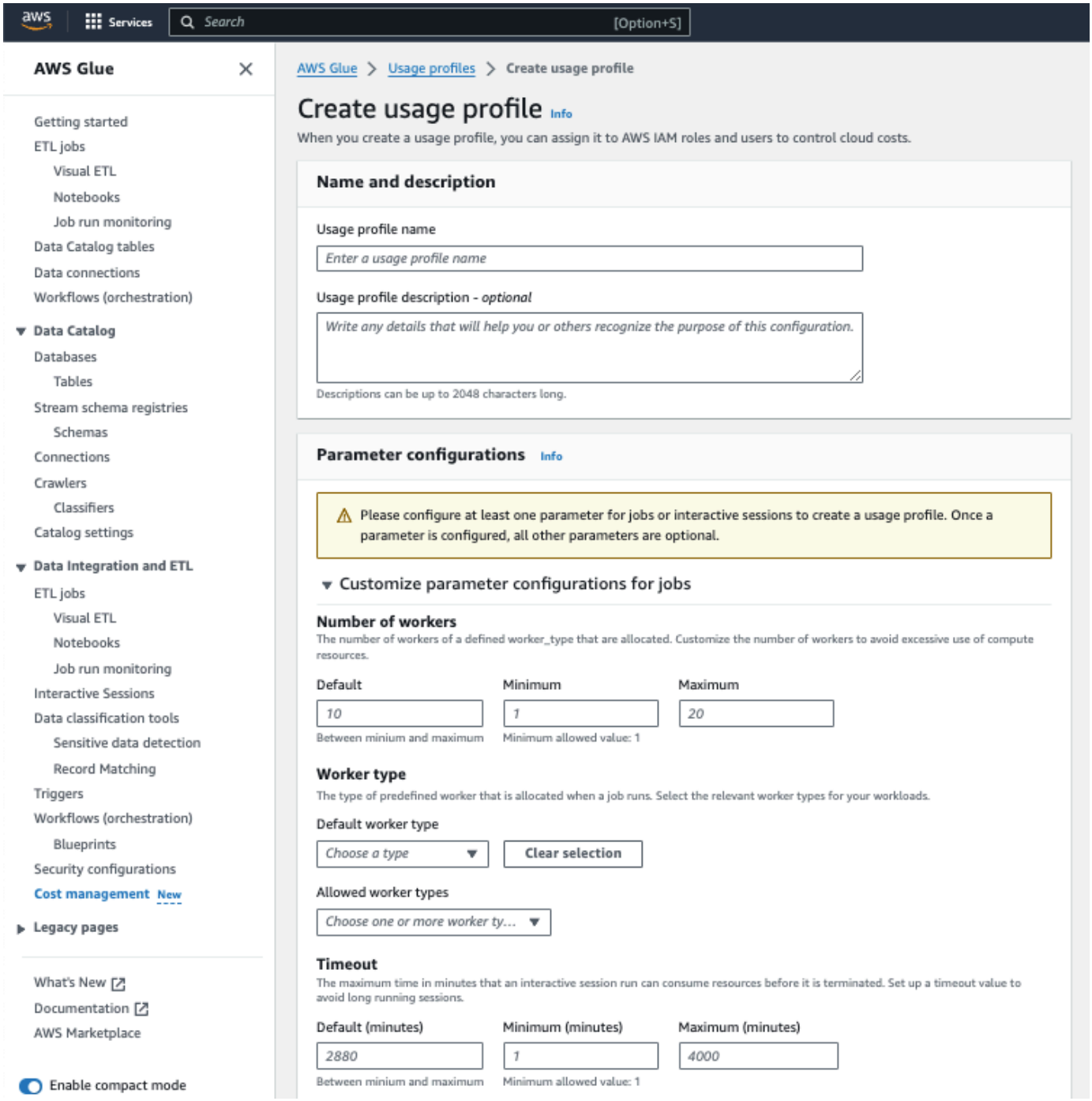
- Timeout im Leerlauf — definiert die Anzahl der Minuten, für die eine interaktive Sitzung inaktiv ist, bevor das Timeout nach dem Ausführen einer Zelle überschritten wird. Definieren Sie das Leerlauf-Timeout für interaktive Sitzungen, die nach Abschluss der Arbeit beendet werden sollen. Der Timeout-Bereich für Leerlauf sollte innerhalb der Timeout-Grenze liegen.

  Sie können einen Standard-, Mindest- und Höchstwert in Minuten festlegen. Der Mindestwert ist 1 (Minute). Das AWS Glue Standard-Timeout beträgt zwar 2880 Minuten, Sie können jedoch einen beliebigen Standardwert im Nutzungsprofil festlegen.

  Es hat sich bewährt, einen Wert für „Standard“ festzulegen. Dieser Wert wird für die Sitzungserstellung verwendet, wenn vom Benutzer kein Wert festgelegt wurde.

Um ein AWS Glue Nutzungsprofil als Administrator (Konsole) zu erstellen

1. Wählen Sie im linken Navigationsmenü die Option Kostenmanagement aus.
2. Wählen Sie Nutzungsprofil erstellen.
3. Geben Sie den Namen des Nutzungsprofils für das Nutzungsprofil ein.
4. Geben Sie optional eine Beschreibung ein, anhand derer andere den Zweck des Nutzungsprofils erkennen können.
5. Definieren Sie mindestens einen Parameter im Profil. Jedes Feld im Formular ist ein Parameter. Zum Beispiel das Mindestzeitlimit für den Leerlauf der Sitzung.
6. Definieren Sie alle optionalen Tags, die für das Nutzungsprofil gelten.
7. Wählen Sie Speichern.



So erstellen Sie ein Nutzungsprofil (AWS CLI)

1. Geben Sie den folgenden Befehl ein.

```
aws glue create-usage-profile --name profile-name --configuration file://
config.json --tags list-of-tags
```

wo config.json Parameterwerte für interaktive Sessions (`SessionConfiguration`) und Jobs (`JobConfiguration`) definieren kann:

```
//config.json (There is a separate blob for session/job configuration
{
    "SessionConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
        },
        "idleTimeout": {
            "DefaultValue": "30",
            "MinValue": "10",
            "MaxValue": "4000"
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.1X",
                "G.2X",
                "G.4X",
                "G.8X",
                "G.12X",
                "G.16X",
                "R.1X",
                "R.2X",
                "R.4X",
                "R.8X"
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
        }
    },
    "JobConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
```

```
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.1X",
                "G.2X",
                "G.4X",
                "G.8X",
                "G.12X",
                "G.16X",
                "R.1X",
                "R.2X",
                "R.4X",
                "R.8X"
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
        }
    }
}
```

2. Geben Sie den folgenden Befehl ein, um das erstellte Nutzungsprofil zu sehen:

```
aws glue get-usage-profile --name profile-name
```

Die Antwort:

```
{
    "ProfileName": "foo",
    "Configuration": {
        "SessionConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.1X",
```

```
                    "G.2X",
                    "G.4X",
                    "G.8X",
                    "G.12X",
                    "G.16X",
                    "R.1X",
                    "R.2X",
                    "R.4X",
                    "R.8X"
                ]
            },
            "timeout": {
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            },
            "idleTimeout": {
                "DefaultValue": "30",
                "MinValue": "10",
                "MaxValue": "4000"
            }
        },
        "JobConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.1X",
                    "G.2X",
                    "G.4X",
                    "G.8X",
                    "G.12X",
                    "G.16X",
                    "R.1X",
                    "R.2X",
                    "R.4X",
                    "R.8X"
                ]
            },
            "timeout": {
```

```
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            }
        }
    },
    "CreatedOn": "2024-01-19T23:15:24.542000+00:00"
}
```

Zusätzliche CLI-Befehle zur Verwaltung von Nutzungsprofilen:

- als Klebstoff list-usage-profiles
- aws glue update-usage-profile --name --configuration *profile-name file://config.json*
- aws glue --name delete-usage-profile *profile-name*

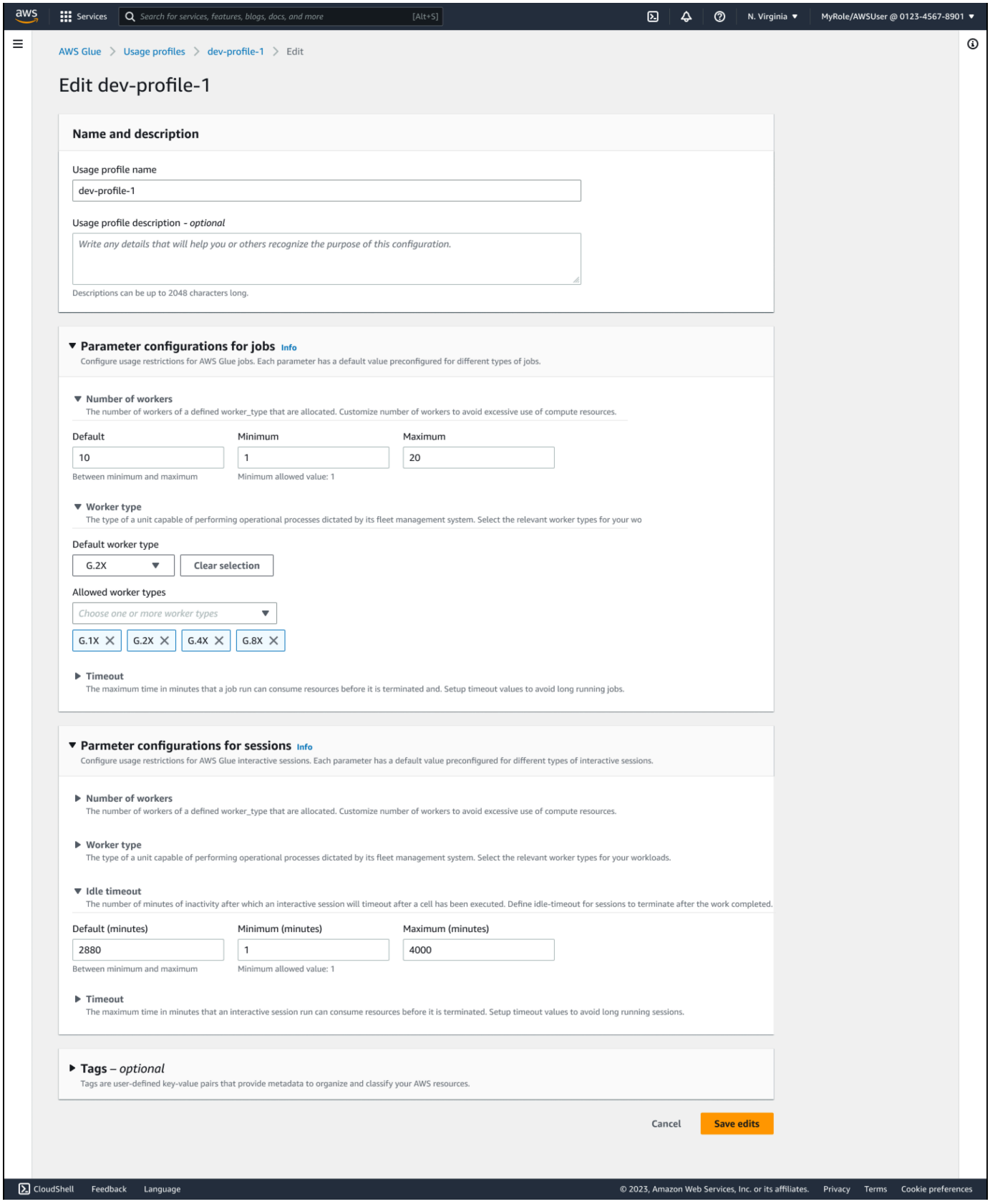
Ein Nutzungsprofil bearbeiten

Administratoren können von ihnen erstellte Nutzungsprofile bearbeiten, um die Profilparameterwerte für Jobs und interaktive Sitzungen zu ändern.

Um ein Nutzungsprofil zu bearbeiten:

Um ein AWS Glue Nutzungsprofil als Administrator (Konsole) zu bearbeiten

1. Wählen Sie im linken Navigationsmenü die Option Kostenmanagement aus.
2. Wählen Sie ein Nutzungsprofil aus, für dessen Bearbeitung Sie berechtigt sind, und wählen Sie Bearbeiten.
3. Nehmen Sie nach Bedarf Änderungen am Profil vor. Standardmäßig werden die Parameter, die bereits Werte haben, erweitert.
4. Wählen Sie Änderungen speichern.


Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
AWS Glue > Usage profiles > dev-profile-1 > Edit
Edit dev-profile-1
Name and description
Usage profile name
dev-profile-1
Usage profile description - optional
Write any details that will help you or others recognize the purpose of this configuration.
Descriptions can be up to 2048 characters long.
Parameter configurations for jobs Info
Configure usage restrictions for AWS Glue jobs. Each parameter has a default value preconfigured for different types of jobs.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Default
10
Between minimum and maximum
Minimum
1
Minimum allowed value: 1
Maximum
20
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your wo
Default worker type
G.2X
Clear selection
Allowed worker types
Choose one or more worker types
G.1X
G.2X
G.4X
G.8X
Timeout
The maximum time in minutes that a job run can consume resources before it is terminated and. Setup timeout values to avoid long running jobs.
Parmeter configurations for sessions Info
Configure usage restrictions for AWS Glue interactive sessions. Each parameter has a default value preconfigured for different types of interactive sessions.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your workloads.
Idle timeout
The number of minutes of inactivity after which an interactive session will timeout after a cell has been executed. Define idle-timeout for sessions to terminate after the work completed.
Default (minutes)
2880
Between minimum and maximum
Minimum (minutes)
1
Minimum allowed value: 1
Maximum (minutes)
4000
Timeout
The maximum time in minutes that an interactive session run can consume resources before it is terminated. Setup timeout values to avoid long running sessions.
Tags – optional
Tags are user-defined key-value pairs that provide metadata to organize and classify your AWS resources.
Cancel
Save edits
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

So bearbeiten Sie ein Nutzungsprofil (AWS CLI)

- Geben Sie den folgenden Befehl ein. Es wird dieselbe `--configuration` Dateisyntax verwendet wie oben im Befehl create gezeigt.

  ```
  aws glue update-usage-profile --name profile-name --configuration file://
  config.json
  ```

  wobei config.json Parameterwerte für interaktive Sessions (`SessionConfiguration`) und Jobs (`JobConfiguration`) definiert:

Zuweisen eines Nutzungsprofils

In der Spalte Nutzungsstatus auf der Seite Verwendungsprofile wird angezeigt, ob Benutzern ein Nutzungsprofil zugewiesen ist. Wenn Sie den Mauszeiger über den Status bewegen, werden die zugewiesenen IAM-Entitäten angezeigt.

Der Administrator kann Personen, die Ressourcen erstellen users/roles AWS Glue , ein AWS Glue Nutzungsprofil zuweisen. Das Zuweisen eines Profils ist eine Kombination aus zwei Aktionen:

- Anschließend wird das user/role IAM-Tag mit dem `glue:UsageProfile` Schlüssel aktualisiert
- Aktualisierung der IAM-Richtlinie des Benutzers/der Rolle.

Für Benutzer, die AWS Glue Studio zum Erstellen von jobs/interactive Sitzungen verwenden, kennzeichnet der Administrator die folgenden Rollen:

- Bei Einschränkungen bei Aufträgen markiert der Administrator die Konsolenrolle, bei der der Benutzer angemeldet ist
- Bei Einschränkungen interaktiver Sitzungen markiert der Administrator die Rolle, die der Benutzer bei der Erstellung des Notizbuchs angegeben hat

Im Folgenden finden Sie ein Beispiel für eine Richtlinie, die der Administrator für den IAM aktualisieren muss, der users/roles die AWS Glue Ressourcen erstellt:

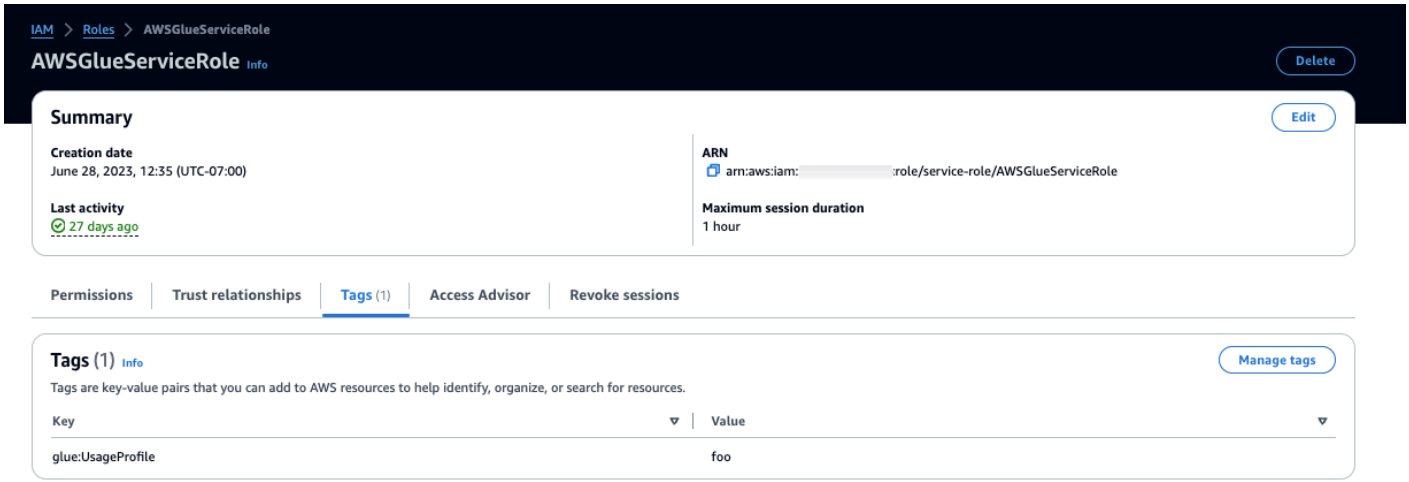
```
{
    "Effect": "Allow",
    "Action": [
        "glue:GetUsageProfile"
```

```
    ],
    "Resource": [
        "arn:aws:glue:us-east-1:123456789012:usageProfile/foo"
    ]
}
```

AWS Glue validiert Anfragen für Aufträge, Auftragsausführungen und Sitzungen auf der Grundlage der im AWS Glue Nutzungsprofil angegebenen Werte und löst eine Ausnahme aus, wenn die Anforderung nicht zulässig ist. Bei synchroner APIs Ausführung wird dem Benutzer ein Fehler gemeldet. Bei asynchronen Pfaden wird eine fehlgeschlagene Auftragsausführung mit der Fehlermeldung erstellt, dass der Eingabeparameter außerhalb des zulässigen Bereichs für das zugewiesene Profil des Benutzers/der Rolle liegt.

Um einem Benutzer/einer Rolle ein Nutzungsprofil zuzuweisen:

1. Öffnen Sie die IAM-Konsole (Identity and Access Management).
2. Wählen Sie in der linken Navigationsleiste Benutzer oder Rollen aus.
3. Wählen Sie einen Benutzer oder eine Rolle aus.
4. Wählen Sie die Registerkarte Tags aus.
5. Wählen Sie Neues Tag hinzufügen
6. Fügen Sie ein Tag mit dem Schlüssel `glue:UsageProfile` und dem Wert des Namens Ihres Nutzungsprofils hinzu.
7. Wählen Sie Save Changes (Änderungen speichern)



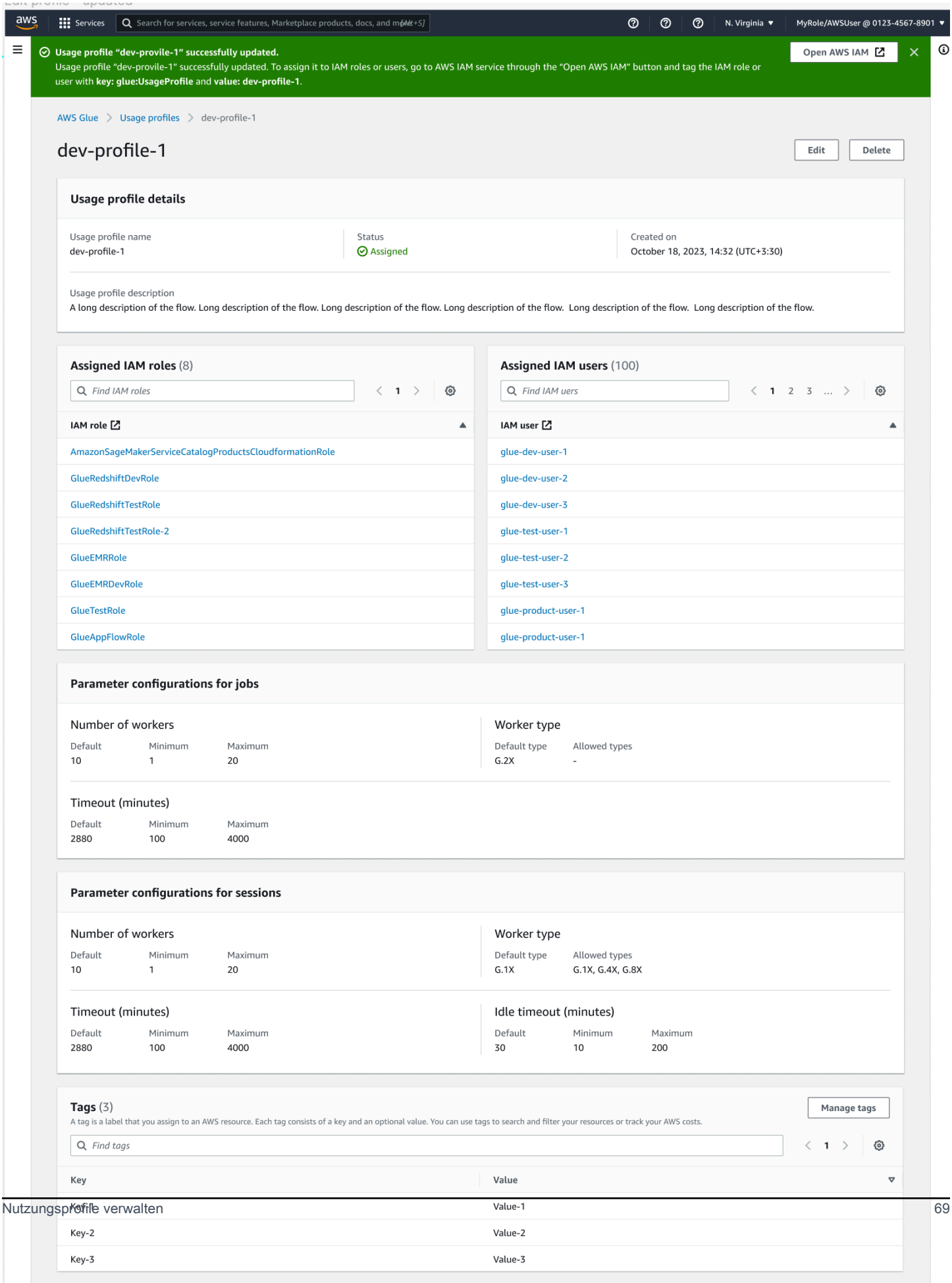
Ihr zugewiesenes Nutzungsprofil anzeigen

Benutzer können ihre zugewiesenen Nutzungsprofile einsehen und sie verwenden, wenn sie API-Aufrufe tätigen, um AWS Glue Job- und Sitzungsressourcen zu erstellen oder einen Job zu starten.

Profilberechtigungen werden in IAM-Richtlinien bereitgestellt. Solange die Anruferrichtlinie über die `glue:UsageProfile` entsprechende Berechtigung verfügt, kann ein Benutzer das Profil sehen. Andernfalls erhalten Sie die Fehlermeldung Zugriff verweigert.

So zeigen Sie ein zugewiesenes Nutzungsprofil an:

1. Wählen Sie im linken Navigationsmenü die Option Kostenmanagement aus.
2. Wählen Sie ein Nutzungsprofil aus, zu dessen Anzeige Sie berechtigt sind.


Services
Search for services, service features, Marketplace products, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
Usage profile "dev-provile-1" successfully updated.
Usage profile "dev-provile-1" successfully updated. To assign it to IAM roles or users, go to AWS IAM service through the "Open AWS IAM" button and tag the IAM role or user with key: glue:UsageProfile and value: dev-profile-1.
Open AWS IAM
AWS Glue > Usage profiles > dev-profile-1
dev-profile-1
Edit
Delete
Usage profile details
Usage profile name
dev-profile-1
Status
Assigned
Created on
October 18, 2023, 14:32 (UTC+3:30)
Usage profile description
A long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow.
Assigned IAM roles (8)
Find IAM roles
IAM role
AmazonSageMakerServiceCatalogProductsCloudformationRole
GlueRedshiftDevRole
GlueRedshiftTestRole
GlueRedshiftTestRole-2
GlueEMRRole
GlueEMRDevRole
GlueTestRole
GlueAppFlowRole
Assigned IAM users (100)
Find IAM uers
IAM user
glue-dev-user-1
glue-dev-user-2
glue-dev-user-3
glue-test-user-1
glue-test-user-2
glue-test-user-3
glue-product-user-1
glue-product-user-1
Parameter configurations for jobs
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.2X
Allowed types -
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Parameter configurations for sessions
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.1X
Allowed types G.1X, G.4X, G.8X
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Idle timeout (minutes)
Default 30
Minimum 10
Maximum 200
Tags (3)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Manage tags
Find tags
Key
Value
Key-1
Value-1
Key-2
Value-2
Key-3
Value-3

Nutzungsprofile und Jobs

Erstellen von Jobs mit Nutzungsprofilen

Beim Erstellen von Aufträgen gelten die in Ihrem Nutzungsprofil festgelegten Grenzwerte und Standardeinstellungen. Ihr Profil wird dem Job beim Speichern zugewiesen.

Jobs mit Nutzungsprofilen ausführen

Wenn Sie eine Auftragsausführung starten, werden die im AWS Glue Profil Ihres Anrufers festgelegten Grenzwerte durchgesetzt. Wenn es keinen direkten Anrufer gibt, wendet Glue dann die Limits aus dem Profil an, das der Autor dem Job zugewiesen hat.

> **Note**
>
> Wenn ein Job nach einem Zeitplan ausgeführt wird (durch AWS Glue Workflows oder AWS Glue Trigger), wird das dem Job zugewiesene Profil angewendet, das der Autor dem Job zugewiesen hat.
> Wenn ein Job von einem externen Dienst (Step Functions, MWAA) oder einer `StartJobRun` API ausgeführt wird, wird das Profillimit des Anrufers durchgesetzt.

Für AWS Glue Workflows oder AWS Glue Trigger: Bereits bestehende Jobs müssen aktualisiert werden, um den neuen Profilnamen zu speichern, sodass die Profilgrenzen (Mindest-, Maximal- und zulässige Worker) zur Laufzeit für geplante Läufe durchgesetzt werden.

Ein für Jobs zugewiesenes Nutzungsprofil anzeigen

Das Ihren Jobs zugewiesene Profil (das zur Laufzeit mit geplanten AWS Glue Workflows oder AWS Glue Triggern verwendet wird) finden Sie auf der Registerkarte Job-Details. Sie können sich auch das Profil ansehen, das in früheren Läufen verwendet wurde, auf der Registerkarte mit den Details zu den Auftragsausführungen.

Aktualisieren oder Löschen eines an einen Auftrag angehängten Nutzungsprofils

Das einem Job zugewiesene Profil wird bei der Aktualisierung geändert. Wenn dem Autor kein Nutzungsprofil zugewiesen wurde, werden alle zuvor mit dem Job verknüpften Profile aus dem Job entfernt.

Erste Schritte mit dem AWS Glue Data Catalog

Das AWS Glue Data Catalog ist Ihr persistenter Speicher für technische Metadaten. Es handelt sich um einen verwalteten Dienst, den Sie zum Speichern, Kommentieren und Teilen von Metadaten in der AWS Cloud verwenden können. Weitere Informationen finden Sie unter AWS Glue Data Catalog.

Das Tool AWS Glue Die Konsole und einige Benutzeroberflächen wurden kürzlich aktualisiert.

Übersicht

Sie können dieses Tutorial verwenden, um Ihr erstes zu erstellen AWS Glue Datenkatalog, der einen Amazon S3 S3-Bucket als Datenquelle verwendet.

In diesem Tutorial gehen Sie wie folgt vor, indem Sie AWS Glue Konsole:

1. Erstellen einer Datenbank.
2. Erstellen einer Tabelle
3. Verwenden eines Amazon-S3-Buckets als Datenquelle

Nachdem Sie diese Schritte abgeschlossen haben, haben Sie erfolgreich einen Amazon S3 S3-Bucket als Datenquelle zum Auffüllen des AWS Glue Datenkatalog.

Schritt 1: Erstellen einer Datenbank

Melden Sie sich zunächst bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole.

Um eine Datenbank mit dem zu erstellen AWS Glue Konsole:

1. Im AWS Glue Konsole, wählen Sie im Menü auf der linken Seite unter Datenkatalog die Option Datenbanken aus.
2. Wählen Sie Add database (Datenbank hinzufügen).
3. Geben Sie auf der Seite „Datenbank erstellen“ einen Namen für die Datenbank ein. Legen Sie im Abschnitt Speicherort – optional den URI-Speicherort für die Verwendung durch Clients des Datenkatalogs fest. Wenn Sie diesen nicht kennen, können Sie mit der Erstellung der Datenbank fortfahren.
4. (Optional). Geben Sie eine Beschreibung für die Datenbank ein.

5. Wählen Sie Datenbank erstellen aus.

Herzlichen Glückwunsch, Sie haben gerade Ihre erste Datenbank mit dem eingerichtet AWS Glue console. Ihre neue Datenbank wird in der Liste der verfügbaren Datenbanken angezeigt. Sie können die Datenbank bearbeiten, indem Sie den Namen der Datenbank aus dem Datenbanken-Dashboard auswählen.

Nächste Schritte

Andere Möglichkeiten, eine Datenbank zu erstellen:

Sie haben gerade eine Datenbank erstellt mit AWS Glue Konsole, aber es gibt auch andere Möglichkeiten, eine Datenbank zu erstellen:

- Sie können Crawler verwenden, um automatisch eine Datenbank und Tabellen für Sie zu erstellen. Informationen zum Einrichten einer Datenbank mithilfe von Crawlern finden Sie unter Arbeiten mit Crawlern im AWS Glue Konsole.
- Sie können AWS CloudFormation Vorlagen verwenden. Siehe Erstellen AWS Glue Ressourcen mithilfe von AWS Glue Data Catalog Vorlagen.
- Sie können eine Datenbank auch mit dem erstellen AWS Glue Datenbank-API-Operationen.

  Um eine Datenbank mit der `create`-Operation zu erstellen, strukturieren Sie die Anforderung, indem Sie die (erforderlichen) `DatabaseInput`-Parameter einschließen.

  Beispiel:

  Im Folgenden finden Sie Beispiele, wie Sie die CLI, Boto3 oder DDL verwenden können, um eine Tabelle basierend auf derselben Datei flights_data.csv aus dem S3-Bucket zu definieren, den Sie im Tutorial verwendet haben.

  CLI

  ```
  aws glue create-database --database-input "{\"Name\":\"clidb\"}"
  ```

  Boto3

  ```
  glueClient = boto3.client('glue')
  ```

```
response = glueClient.create_database(
    DatabaseInput={
        'Name': 'boto3db'
    }
)
```

Weitere Informationen zu den Datentypen, -Strukturen und -Vorgängen der Datenbank-API finden Sie unter Datenbank-API.

Nächste Schritte

Im nächsten Abschnitt erstellen Sie eine Tabelle und fügen diese Tabelle Ihrer Datenbank hinzu.

Sie können sich auch die Einstellungen und Berechtigungen für Ihren Datenkatalog ansehen. Weitere Informationen finden Sie unter Arbeiten mit Datenkatalogeinstellungen im AWS Glue Konsole.

Schritt 2. Erstellen einer Tabelle

In diesem Schritt erstellen Sie eine Tabelle mit AWS Glue console.

1. Im AWS Glue Konsole, wählen Sie im Menü auf der linken Seite Tabellen.
2. Wählen Sie Add table (Tabelle hinzufügen).
3. Richten Sie die Eigenschaften Ihrer Tabelle ein, indem Sie einen Namen für Ihre Tabelle in Table details (Tabellendetails) eingeben.
4. Wählen Sie im Abschnitt Databases (Datenbanken) die Datenbank aus dem Dropdown-Menü aus, die Sie in Schritt 1 erstellt haben.
5. Im Abschnitt Add a data store (Datenspeicher hinzufügen) wird S3 standardmäßig als Quelltyp ausgewählt.
6. Wählen Sie im Abschnitt Data is located in (Daten befinden sich in) die Option Specified path in another account (Angegebener Pfad in einem anderen Konto) aus.
7. Kopieren Sie den Pfad für das Eingabefeld Include path (Pfad einschließen) und fügen Sie ihn ein:

   ```
   s3://crawler-public-us-west-2/flight/2016/csv/
   ```
8. Wählen Sie im Abschnitt Data format (Datenformat) für Classification (Klassifizierung) CSV und für Delimiter (Trennzeichen) comma (,) (Komma) aus. Wählen Sie Weiter.

9. Sie werden aufgefordert, ein Schema zu definieren. Ein Schema definiert die Struktur und das Format eines Datensatzes. Wählen Sie Add column (Spalte hinzufügen). (Weitere Informationen finden Sie unter Schema Registries)
10. Geben Sie die Spalteneigenschaften an:

    a. Geben Sie einen Spaltennamen ein.
    b. Als Spaltentyp ist standardmäßig bereits „Zeichenfolge“ ausgewählt.
    c. Als Spaltennummer ist standardmäßig bereits „1“ ausgewählt.
    d. Wählen Sie Hinzufügen aus.
11. Sie werden aufgefordert, Partitionsindizes hinzuzufügen. Dieser Schritt ist optional. Um diesen Schritt zu überspringen, wählen Sie Next (Weiter).
12. Eine Zusammenfassung der Tabelleneigenschaften wird angezeigt. Wenn alles wie erwartet aussieht, wählen Sie Create. Wählen Sie andernfalls Back (Zurück) und nehmen Sie nach Bedarf Änderungen vor.

Herzlichen Glückwunsch, Sie haben erfolgreich eine Tabelle manuell erstellt und mit einer Datenbank verknüpft. Ihre neu erstellte Tabelle wird im Tabellen-Dashboard angezeigt. Über das Dashboard können Sie alle Ihre Tabellen ändern und verwalten.

Weitere Informationen finden Sie unter Arbeiten mit Tabellen im AWS Glue Konsole.

Nächste Schritte

Nächste Schritte

Jetzt, da der Datenkatalog gefüllt ist, können Sie mit der Erstellung von Aufträgen beginnen AWS Glue. Weitere Informationen finden Sie unter Visual ETL-Jobs erstellen mit AWS Glue Studio.

Neben der Verwendung der Konsole gibt es noch andere Möglichkeiten, Tabellen im Datenkatalog zu definieren, darunter:

- Erstellen und Ausführen eines Crawlers
- Hinzufügen von Klassifikatoren zu einem Crawler in AWS Glue
- Mit dem AWS Glue Tabellen-API
- Verwenden der AWS Glue Data Catalog -Vorlage
- Migrieren eines Apache Hive-Metastores
- Verwenden der AWS CLI Boto3- oder Datendefinitionssprache (DDL)

Im Folgenden finden Sie Beispiele, wie Sie die CLI, Boto3 oder DDL verwenden können, um eine Tabelle basierend auf derselben Datei flights_data.csv aus dem S3-Bucket zu definieren, den Sie im Tutorial verwendet haben.

Informationen zur Strukturierung eines Befehls finden Sie in der Dokumentation. AWS CLI Das CLI-Beispiel enthält die JSON-Syntax für den Wert „aws glue create-table --table-input“.

CLI

```
{
        "Name": "flights_data_cli",
        "StorageDescriptor": {
            "Columns": [
                {
                    "Name": "year",
                    "Type": "bigint"
                },
                {
                    "Name": "quarter",
                    "Type": "bigint"
                }
            ],
            "Location": "s3://crawler-public-us-west-2/flight/2016/csv",
            "InputFormat": "org.apache.hadoop.mapred.TextInputFormat",
            "OutputFormat":
 "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
            "Compressed": false,
            "NumberOfBuckets": -1,
            "SerdeInfo": {
                "SerializationLibrary":
 "org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "Parameters": {
                    "field.delim": ",",
                    "serialization.format": ","
                }
            }
        },
        "PartitionKeys": [
            {
                "Name": "mon",
                "Type": "string"
            }
```

```
        ],
        "TableType": "EXTERNAL_TABLE",
        "Parameters": {
            "EXTERNAL": "TRUE",
            "classification": "csv",
            "columnsOrdered": "true",
            "compressionType": "none",
            "delimiter": ",",
            "skip.header.line.count": "1",
            "typeOfData": "file"
        }
    }
```

Boto3

```
import boto3

glue_client = boto3.client("glue")

response = glue_client.create_table(
    DatabaseName='sampledb',
    TableInput={
        'Name': 'flights_data_manual',
    'StorageDescriptor': {
      'Columns': [{
        'Name': 'year',
        'Type': 'bigint'
      }, {
        'Name': 'quarter',
        'Type': 'bigint'
      }],
      'Location': 's3://crawler-public-us-west-2/flight/2016/csv',
      'InputFormat': 'org.apache.hadoop.mapred.TextInputFormat',
      'OutputFormat':
 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat',
      'Compressed': False,
      'NumberOfBuckets': -1,
      'SerdeInfo': {
        'SerializationLibrary':
 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe',
        'Parameters': {
          'field.delim': ',',
```

```
          'serialization.format': ','
        }
      },
    },
    'PartitionKeys': [{
      'Name': 'mon',
      'Type': 'string'
    }],
    'TableType': 'EXTERNAL_TABLE',
    'Parameters': {
      'EXTERNAL': 'TRUE',
      'classification': 'csv',
      'columnsOrdered': 'true',
      'compressionType': 'none',
      'delimiter': ',',
      'skip.header.line.count': '1',
      'typeOfData': 'file'
    }
    }
)
```

DDL

```
CREATE EXTERNAL TABLE `sampledb`.`flights_data` (
  `year` bigint,
  `quarter` bigint)
PARTITIONED BY (
  `mon` string)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ','
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://crawler-public-us-west-2/flight/2016/csv/'
TBLPROPERTIES (
  'classification'='csv',
  'columnsOrdered'='true',
  'compressionType'='none',
  'delimiter'=',',
  'skip.header.line.count'='1',
```

```
  'typeOfData'='file')
```

Netzwerkzugriff auf Datenspeicher einrichten

Um Ihre ETL-Jobs (Extrahieren, Transformieren und Laden) auszuführen, AWS Glue muss auf Ihre Datenspeicher zugreifen können. Wenn ein Auftrag nicht in Ihrem Virtual Private Cloud (VPC)-Subnetz ausgeführt werden muss, z. B. bei der Umwandlung von Daten aus Amazon S3 in Amazon S3, ist keine zusätzliche Konfiguration erforderlich.

Wenn ein Job in Ihrem VPC-Subnetz ausgeführt werden muss, z. B. die Transformation von Daten aus einem JDBC-Datenspeicher in einem privaten Subnetz —AWS Glue richtet elastische Netzwerkschnittstellen ein, die es Ihren Jobs ermöglichen, sich sicher mit anderen Ressourcen innerhalb Ihrer VPC zu verbinden. Jeder Elastic Network-Schnittstelle wird eine private IP-Adresse aus dem IP-Adressbereich innerhalb des von Ihnen angegebenen Subnetzes zugeordnet. Es werden keine öffentlichen IP-Adressen zugeordnet. Sicherheitsgruppen, die in der angegeben sind AWS Glue Verbindungen werden auf jede der Elastic Network-Schnittstellen angewendet. Weitere Informationen finden Sie unter Einrichtung von Amazon VPC für JDBC-Verbindungen zu Amazon RDS-Datenspeichern von AWS Glue.

Alle JDBC-Datenspeicher, auf die der Auftrag zugreift, müssen über das VPC-Subnetz verfügbar sein. Für den Zugriff auf Amazon S3 von Ihrer VPC ist ein VPC-Endpunkt erforderlich. Wenn Ihr Auftrag sowohl auf VPC-Ressourcen als auch auf das öffentliche Internet zugreifen muss, muss der -VPC über ein NAT-Gateway (Network Address Translation) innerhalb des VPC verfügen.

Ein Auftrag oder Entwicklungsendpunkt kann jeweils nur auf eine VPC (und Subnetz) zugreifen. Wenn Sie auf verschiedene Datenspeicher zugreifen müssen VPCs, haben Sie die folgenden Optionen:

- Verwenden Sie für den Zugriff auf die Datenspeicher VPC Peering. Weitere Informationen zu VPC Peering finden Sie unter VPC Peering-Grundlagen
- Verwenden Sie einen Amazon-S3-Bucket als Vermittlungsspeicherort. Trennen Sie die Arbeit in zwei Aufträge mit der Amazon-S3-Ausgabe von Auftrag 1 als Eingabe für Auftrag 2.

Einzelheiten zum Herstellen einer Verbindung zu einem Amazon-Redshift-Datenspeicher mithilfe von Amazon VPC finden Sie unter the section called “Redshift konfigurieren”.

Einzelheiten zum Herstellen einer Verbindung zu Amazon-RDS-Datenspeichern mithilfe von Amazon VPC finden Sie unter the section called “Amazon VPC für die Verbindung mit Amazon RDS-Datenspeichern einrichten”.

Sobald die erforderlichen Regeln in Amazon VPC festgelegt sind, erstellen Sie eine Verbindung in AWS Glue mit den erforderlichen Eigenschaften, um eine Verbindung zu Ihren Datenspeichern herzustellen. Weitere Informationen über die Verbindung finden Sie unter Herstellen einer Verbindung zu Daten.

> **Note**
>
> Stellen Sie sicher, dass Sie Ihre DNS-Umgebung einrichten für AWS Glue. Weitere Informationen finden Sie unterEinrichten des DNS in Ihrer VPC.

Themen

- Einrichten einer VPC zum Herstellen einer Verbindung mit PyPI für AWS Glue
- Einrichten des DNS in Ihrer VPC

Einrichten einer VPC zum Herstellen einer Verbindung mit PyPI für AWS Glue

Der Python Package Index (PyPI) ist ein Repository mit Software für die Programmiersprache Python. Dieses Thema befasst sich mit den Details, die zur Unterstützung der Verwendung von pip-installierten Paketen erforderlich sind (wie vom Sitzungsersteller mithilfe des `--additional-python-modules`-Flags angegeben).

Die Verwendung AWS Glue interaktiver Sitzungen mit einem Connector führt zur Verwendung des VPC-Netzwerks über das für den Connector angegebene Subnetz. Folglich sind AWS Dienste und andere Netzwerkziele nur verfügbar, wenn Sie eine spezielle Konfiguration einrichten.

Zu den Lösungen für dieses Problem gehören:

- Nutzung eines Internet-Gateways, das für Ihre Sitzung erreichbar ist.
- Einrichten und Verwenden eines S3-Buckets mit einem PyPI/einfachen Repository, das den transitiven Abschluss der Abhängigkeiten eines Paketsatzes enthält.

- Verwendung eines CodeArtifact Repositorys, das PyPI spiegelt und an Ihre VPC angeschlossen ist.

Einrichten eines Internet-Gateways

Die technischen Aspekte werden in den NAT-Gateway-Anwendungsfällen detailliert beschrieben. Beachten Sie jedoch die folgenden Anforderungen für die Verwendung von `--additional-python-modules`. Insbesondere erfordert `--additional-python-modules` Zugriff auf pypi.org, was durch die Konfiguration Ihrer VPC bestimmt wird. Beachten Sie die folgenden Voraussetzungen:

1. Die Anforderung, zusätzliche Python-Module über pip install für eine Benutzersitzung zu installieren. Wenn die Sitzung einen Konnektor verwendet, kann dies Auswirkungen auf Ihre Konfiguration haben.
2. Wenn ein Konnektor mit `--additional-python-modules` verwendet wird, muss beim Starten der Sitzung das mit dem `PhysicalConnectionRequirements` des Konnektor verknüpfte Subnetz einen Netzwerkpfad für die Verbindung zu pypi.org bereitstellen.
3. Sie müssen feststellen, ob Ihre Konfiguration korrekt ist oder nicht.

Einrichten eines Amazon-S3-Buckets zum Hosten eines gezielten PyPI/einfachen Repositorys

In diesem Beispiel wird in Amazon S3 ein PyPI-Spiegel für eine Reihe von Paketen und deren Abhängigkeiten eingerichtet.

So richten Sie den PyPI-Spiegel für eine Reihe von Paketen ein:

```
# pip download all the dependencies
pip download -d s3pypi --only-binary :all: plotly gglplot
pip download -d s3pypi --platform manylinux_2_17_x86_64 --only-binary :all: psycopg2-
binary
# create and upload the pypi/simple index and wheel files to the s3 bucket
s3pypi -b test-domain-name --put-root-index -v s3pypi/*
```

Wenn Sie bereits über ein vorhandenes Artefakt-Repository verfügen, verfügt dieses über eine Index-URL zur Verwendung durch pip, die Sie anstelle der Beispiel-URL für den Amazon-S3-Bucket wie oben angeben können.

So verwenden Sie die benutzerdefinierte Index-URL mit einigen Beispielpaketen:

```
%%configure
{
    "--additional-python-modules": "psycopg2_binary==2.9.5",
    "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
}
```

Einen CodeArtifact Pypi-Spiegel einrichten, der an Ihre VPC angeschlossen ist

So richten Sie einen Spiegel ein:

1. Erstellen Sie ein Repository in derselben Region wie das vom Konnektor verwendete Subnetz.

   Wählen Sie `Public upstream repositories` und klicken Sie auf `pypi-store`.
2. Gewähren Sie Zugriff auf das Repository von der VPC für das Subnetz.
3. Geben Sie die richtige `--index-url` mithilfe des `python-modules-installer-option` an.

   ```
   %%configure
   {
       "--additional-python-modules": "psycopg2_binary==2.9.5",
       "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
   test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
   }
   ```

Weitere Informationen finden Sie unter Verwendung CodeArtifact von einer VPC aus.

Einrichten des DNS in Ihrer VPC

Domain Name System (DNS) ist ein Standard, nach dem Namen, die im Internet verwendet werden, entsprechend der zugehörigen IP-Adressen aufgelöst werden. Ein DNS-Hostname benennt einen Computer eindeutig und besteht aus einem Hostnamen und einem Domainnamen. DNS-Server lösen DNS-Hostnamen zu den entsprechenden IP-Adressen auf.

Zum Einrichten des DNS in Ihrer VPC müssen Sie sicherstellen, dass sowohl DNS-Hostnamen als auch die DNS-Auflösung in Ihrer VPC aktiviert sind. Die VPC-Netzwerkattribute `enableDnsHostnames` und `enableDnsSupport` müssen auf `true` gesetzt werden. Zum Anzeigen und Ändern dieser Attribute wechseln Sie über https://console.aws.amazon.com/vpc/ zur VPC-Konsole.

Weitere Informationen finden Sie unter Verwenden von DNS in Ihrer VPC. Sie können auch den modify-vpc-attributeBefehl AWS CLI und aufrufen, um die VPC-Netzwerkattribute zu konfigurieren.

> **Note**
>
> Stellen Sie bei Verwendung von Route 53 sicher, dass ihre Konfiguration keine DNS-Netzwerkattribute überschreibt.

Einrichten der Verschlüsselung in AWS Glue

Das folgende Beispiel zeigt die Optionen, die Sie konfigurieren müssen, wenn Sie die Verschlüsselung mit AWS Glue verwenden. Das Beispiel zeigt die Verwendung bestimmter AWS Key Management Service (AWS KMS) -Schlüssel, aber Sie können je nach Ihren speziellen Bedürfnissen auch andere Einstellungen wählen. Dieser Workflow hebt nur die Optionen hervor, die sich auf die Verschlüsselung bei der Einrichtung von AWS Glue beziehen.

1. Wenn der Benutzer der AWS Glue-Konsole keine Berechtigungsrichtlinie verwendet, die alle AWS Glue-API-Operationen (zum Beispiel `"glue:*"`) zulässt, bestätigen Sie, dass die folgenden Berechtigungen erlaubt sind:
   - `"glue:GetDataCatalogEncryptionSettings"`
   - `"glue:PutDataCatalogEncryptionSettings"`
   - `"glue:CreateSecurityConfiguration"`
   - `"glue:GetSecurityConfiguration"`
   - `"glue:GetSecurityConfigurations"`
   - `"glue:DeleteSecurityConfiguration"`
2. Jeder Client, der auf einen verschlüsselten Katalog zugreift, d. h. jeder Konsolenbenutzer, Crawler, Auftrag oder Entwicklungsendpunkt, benötigt die folgenden Berechtigungen.

   JSON

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:GenerateDataKey",
           "kms:Decrypt",
```

```
            "kms:Encrypt"
        ],
        "Resource": "<key-arns-used-for-data-catalog>"
    }
}
```

3. Jeder Benutzer oder jede Rolle, der bzw. die auf ein verschlüsseltes Verbindungspasswort zugreift, benötigt die folgenden Berechtigungen.

   JSON

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:Decrypt"
         ],
     "Resource": "<key-arns-used-for-password-encryption>"
           }
}
```

4. Die Rolle eines jeden Extraktions-, Transformations- und Ladeauftrags (ETL), der verschlüsselte Daten in Amazon S3 schreibt, erfordert die folgenden Berechtigungen.

   JSON

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:Decrypt",
           "kms:Encrypt",
           "kms:GenerateDataKey"
        ],
     "Resource": "<key-arns-used-for-s3>"
    }
}
```

5. Jeder ETL-Job oder Crawler, der verschlüsselte CloudWatch Amazon-Logs schreibt, benötigt die folgenden Berechtigungen in den Schlüssel- und IAM-Richtlinien.

In der Schlüsselrichtlinie (nicht in der IAM-Richtlinie):

```
{
  "Effect": "Allow",
  "Principal": {
   "Service": "logs.region.amazonaws.com"
  },
  "Action": [
   "kms:Encrypt*",
   "kms:Decrypt*",
   "kms:ReEncrypt*",
   "kms:GenerateDataKey*",
   "kms:Describe*"
  ],
  "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
 }
```

Weitere Informationen zu Schlüsselrichtlinien finden Sie unter Verwenden von Schlüsselrichtlinien in AWS KMS im AWS Key Management Service - Entwicklerhandbuch.

Fügen Sie in der IAM-Richtlinie die `logs:AssociateKmsKey`-Berechtigung hinzu:

```
{
  "Effect": "Allow",
  "Principal": {
   "Service": "logs.region.amazonaws.com"
  },
  "Action": [
   "logs:AssociateKmsKey"
  ],
  "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
 }
```

6. Jeder ETL-Auftrag, der ein verschlüsseltes Auftrags-Lesezeichen verwendet, benötigt die folgenden Berechtigungen:

   JSON

   ```
   {
    "Version": "2012-10-17",
     "Statement": {
        "Effect": "Allow",
   ```

```
      "Action": [
            "kms:Decrypt",
            "kms:Encrypt"
       ],
      "Resource": "<key-arns-used-for-job-bookmark-encryption>"
   }
}
```

7. Wählen Sie im Navigationsbereich der AWS Glue-Konsole die Settings (Einstellungen) aus.
   a. Wählen Sie auf der Seite Data catalog settings (Data-Catalog-Einstellungen) die Option Metadata encryption (Metadatenverschlüsselung), um den Data Catalog zu verschlüsseln. Diese Option verschlüsselt alle Objekte im Datenkatalog mit dem von Ihnen ausgewählten AWS KMS Schlüssel.
   b. Wählen Sie für AWS KMS --Schlüssel) die Option aws/glue aus. Sie können auch einen AWS KMS Schlüssel auswählen, den Sie erstellt haben.

   > **Important**
   >
   > AWS Glueunterstützt nur symmetrische Kunden-Masterschlüssel (CMKs). In der Liste der AWS KMS -Schlüssel werden nur symmetrische Schlüssel angezeigt. Wenn Sie jedoch „ AWS KMS Schlüssel-ARN auswählen“ auswählen, können Sie in der Konsole einen ARN für einen beliebigen Schlüsseltyp eingeben. Stellen Sie sicher, dass Sie nur ARNs für symmetrische Schlüssel eingeben.

   Wenn die Verschlüsselung aktiviert ist, muss der Client, der auf den Data Catalog zugreift, über AWS KMS -Berechtigungen verfügen.
8. Wählen Sie im Navigationsbereich Security configurations (Sicherheitskonfigurationen) aus. Eine Sicherheitskonfiguration ist eine Reihe von Sicherheitseigenschaften, die zur Konfiguration von AWS Glue-Prozessen verwendet werden können. Wählen Sie dann Add security configuration (Sicherheitskonfiguration hinzufügen). Wählen Sie in der Konfiguration eine der folgenden Optionen:
   a. Wählen Sie S3 encryption (S3-Verschlüsselung) aus. Für Encryption mode (Verschlüsselungsmodus) wählen Sie SSE-KMS. Wählen Sie für den AWS KMS -Schlüssel aws/s3 (stellen Sie sicher, dass der Benutzer die Berechtigung hat, diesen Schlüssel zu verwenden). Dadurch können Daten, die vom Job in Amazon S3 geschrieben wurden, den AWS verwalteten AWS Glue AWS KMS Schlüssel verwenden.

b. Wählen Sie CloudWatch Protokollverschlüsselung und wählen Sie einen CMK aus. (Stellen Sie sicher, dass der Benutzer über die Berechtigung verfügt, diesen Schlüssel zu verwenden). Weitere Informationen finden Sie unter Verschlüsseln von Protokolldaten in CloudWatch Logs Using AWS KMS im AWS Key Management Service Entwicklerhandbuch.

> **Important**
>
> AWS Glueunterstützt nur symmetrische Kunden-Masterschlüssel ()CMKs. In der Liste der AWS KMS -Schlüssel werden nur symmetrische Schlüssel angezeigt. Wenn Sie jedoch „ AWS KMS Schlüssel-ARN auswählen“ auswählen, können Sie in der Konsole einen ARN für einen beliebigen Schlüsseltyp eingeben. Stellen Sie sicher, dass Sie nur ARNs für symmetrische Schlüssel eingeben.

c. Wählen Sie Advanced properties (Erweiterte Eigenschaften) und aktivieren Sie das Kontrollkästchen Job bookmark encryption (Verschlüsselung von Auftrags-Lesezeichen). Wählen Sie für den AWS KMS -Schlüssel aws/glue (stellen Sie sicher, dass der Benutzer die Berechtigung hat, diesen Schlüssel zu verwenden). Dies ermöglicht die Verschlüsselung von Job-Lesezeichen, die mit dem AWS Glue AWS KMS Schlüssel in Amazon S3 geschrieben wurden.

9. Wählen Sie im Navigationsbereich Connections aus.

   a. Wählen Sie Add connection (Verbindung hinzufügen), um eine Verbindung zum Java Database Connectivity (JDBC)-Datenspeicher herzustellen, der das Ziel Ihres ETL-Auftrags ist.

   b. Um die Verwendung der SSL (Secure Sockets Layer)-Verschlüsselung zu erzwingen, aktivieren Sie Require SSL connection (SSL-Verbindung erforderlich) und testen Sie Ihre Verbindung.

10. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.

    a. Wählen Sie Add job (Auftrag hinzufügen), um einen Auftrag zu erstellen, der Daten transformiert.

    b. Wählen Sie in der Auftragsdefinition die von Ihnen erstellte Sicherheitskonfiguration aus.

11. Führen Sie Ihren Auftrag auf der AWS Glue-Konsole bei Bedarf aus. Stellen Sie sicher, dass alle vom Job geschriebenen Amazon S3 S3-Daten, die vom Job geschriebenen CloudWatch Logs und die Job-Lesezeichen alle verschlüsselt sind.

Einrichtung von Netzwerken für die Entwicklung für AWS Glue

Um Ihre ETL-Skripts (Extrahieren, Transformieren und Laden) auszuführen mit AWS Glue, können Sie Ihre Skripts mithilfe eines Entwicklungsendpunkts entwickeln und testen. Die Verwendung von Entwicklungsendpunkten mit wird nicht unterstützt AWS Glue Jobs der Version 2.0. Für Versionen 2.0 und höher ist die bevorzugte Entwicklungsmethode die Verwendung von Jupyter Notebook mit einem der AWS Glue Kernel. Weitere Informationen finden Sie unter the section called “Erste Schritte mit AWS Glue interaktive Sitzungen”.

Einrichten Ihres Netzwerks für einen Entwicklungsendpunkt

Wenn Sie einen Entwicklungsendpunkt einrichten, geben Sie eine Virtual Private Cloud (VPC), ein Subnetz und Sicherheitsgruppen an.

> **Note**
>
> Stellen Sie sicher, dass Sie Ihre DNS-Umgebung einrichten für AWS Glue. Weitere Informationen finden Sie unterEinrichten des DNS in Ihrer VPC.

Um zu aktivieren AWS Glue Um auf die erforderlichen Ressourcen zuzugreifen, fügen Sie Ihrer Subnetz-Routentabelle eine Zeile hinzu, um dem VPC-Endpunkt eine Präfixliste für Amazon S3 zuzuordnen. Eine Präfixlisten-ID ist zum Erstellen einer ausgehenden Sicherheitsgruppenregel erforderlich, die zulässt, dass Datenverkehr von einer VPC über einen VPC-Endpunkt auf einen AWS -Service zugreift. Zum Vereinfachen der Verbindung mit einem Notebook-Server, der diesem Entwicklungsendpunkt zugewiesen ist, fügen Sie der Routing-Tabelle von Ihrem lokalen Computer aus eine Zeile hinzu, um eine Internet-Gateway-ID anzugeben. Weitere Informationen finden Sie unter VPC Endpoints. Aktualisieren Sie die Subnetzroutentabelle ähnlich der folgenden Tabelle:

Bestimmungsort	Ziel		
10.0.0.0/16	Lokal		
pl-id für Amazon S3	vpce-id		
0.0.0.0/0	igw-xxxx		

Um zu aktivieren AWS Glue Um zwischen den Komponenten zu kommunizieren, geben Sie eine Sicherheitsgruppe mit einer selbstreferenzierenden Regel für eingehende Nachrichten für alle TCP-Ports an. Durch Erstellen einer selbstreferenzierenden Regel können Sie die Quelle auf die gleiche Sicherheitsgruppe in der VPC beschränken und es vermeiden, dass sie für alle Netzwerke offen ist. Die Standardsicherheitsgruppe für Ihre VPC verfügt möglicherweise bereits über eine selbstreferenzierende eingehende Regel für den gesamten Datenverkehr (ALL Traffic).

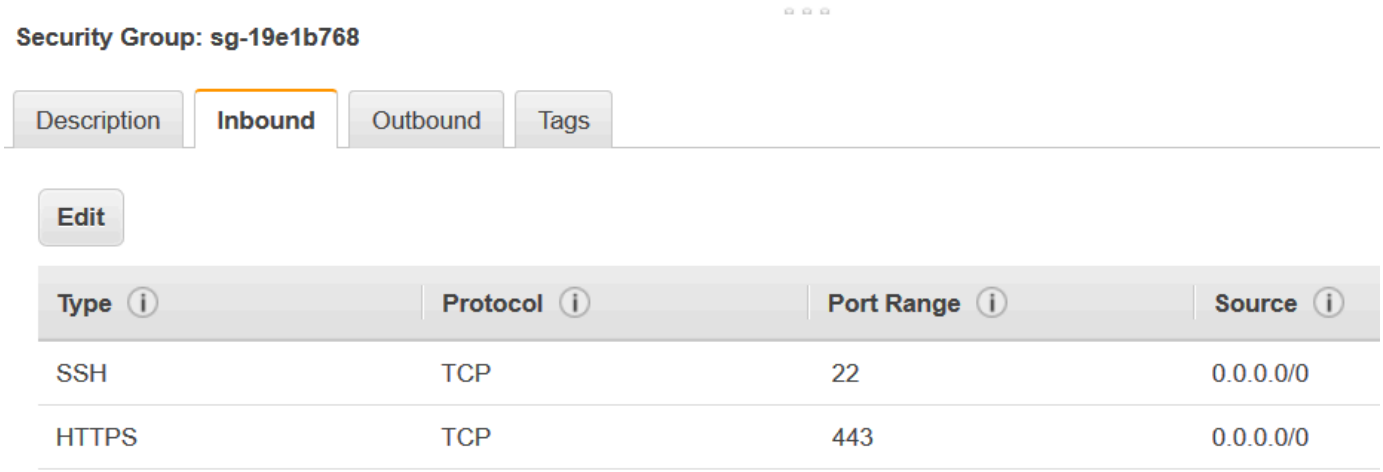
So richten Sie eine Sicherheitsgruppe ein:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die EC2 Amazon-Konsole unter https://console.aws.amazon.com/ec2/.
2. Klicken Sie im linken Navigationsbereich auf Security Groups.
3. Wählen Sie eine vorhandene Sicherheitsgruppe aus der Liste oder die Option Create Security Group (Sicherheitsgruppe erstellen) für die Verwendung mit dem Entwicklungsendpunkt aus.
4. Navigieren Sie im Sicherheitsgruppenbereich zur Registerkarte Eingehend.
5. Fügen Sie eine Regel hinzu, die sich selbst referenziert, um Folgendes zuzulassen AWS Glue Komponenten für die Kommunikation. Insbesondere fügen Sie hinzu oder bestätigen Sie, dass eine Regel des Typs `All TCP` vorhanden ist, das Protokoll `TCP` lautet, der Port-Bereich alle Ports umfasst und deren Quelle über denselben Sicherheitsgruppennamen verfügt wie die Gruppen-ID.

   Die eingehende Regel sollte etwa wie folgt aussehen:

Typ	Protocol (Protokoll)	Port-Bereich	Quelle
Alle TCP	TCP	0–65535	*`security-group`*

   Das nachfolgende Beispiel zeigt eine selbstreferenzierende eingehende Regel:



6. Fügen Sie ebenfalls eine Regel für ausgehenden Datenverkehr hinzu. Öffnen Sie entweder den ausgehenden Datenverkehr für alle Ports oder erstellen eine selbstreferenzierende Regel des Typs `All TCP`, das Protokoll lautet `TCP`, der Port-Bereich umfasst alle Ports und deren Quelle verfügt über denselben Sicherheitsgruppennamen wie die Gruppen-ID.

   Die ausgehende Regel ähnelt einer der folgenden Regeln:

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Alle TCP	TCP	0–65535	*`security-group`*
Gesamter Datenverkehr	ALL	ALL	0.0.0.0/0

Amazon EC2 für einen Notebook-Server einrichten

Mit einem Entwicklungsendpunkt können Sie einen Notebook-Server erstellen, um Ihre ETL-Skripte mit Jupyter Notebooks zu testen. Zum Aktivieren der Kommunikation mit Ihrem Notebook geben Sie eine Sicherheitsgruppe mit Regeln für eingehenden Datenverkehr für HTTPS (Port 443) und SSH (Port 22) an. Stellen Sie sicher, dass die Quelle der Regel entweder 0.0.0.0/0 oder die IP-Adresse des Rechners ist, der die Verbindung mit dem Notebook herstellt.

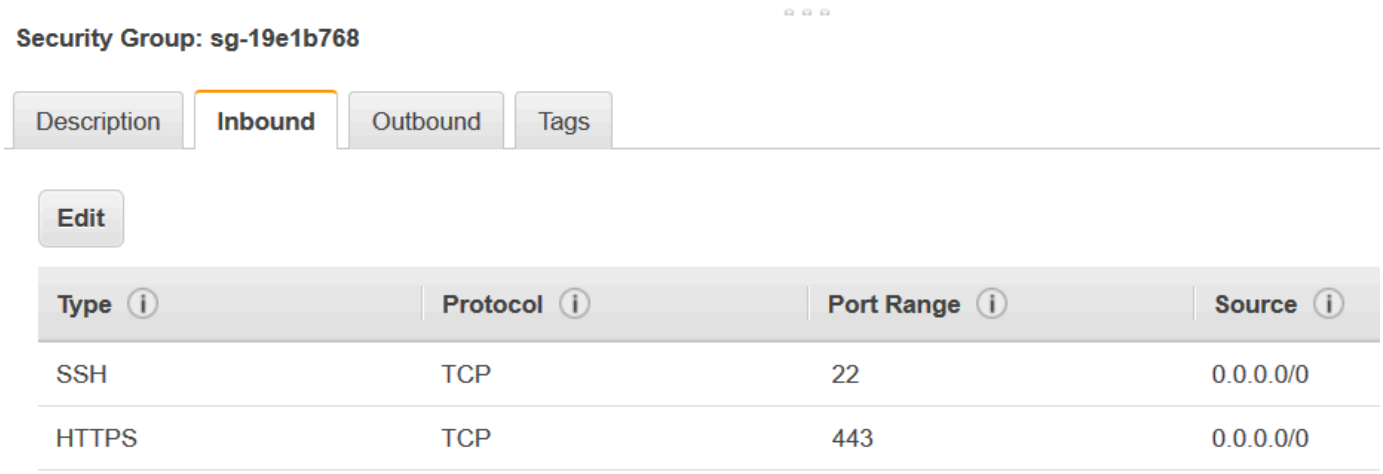
So richten Sie eine Sicherheitsgruppe ein:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die EC2 Amazon-Konsole unter https://console.aws.amazon.com/ec2/.

2. Klicken Sie im linken Navigationsbereich auf Security Groups.
3. Wählen Sie eine vorhandene Sicherheitsgruppe aus der Liste oder die Option Create Security Group (Sicherheitsgruppe erstellen) für die Verwendung mit Ihrem Notebook-Server aus. Die Sicherheitsgruppe, die Ihrem Entwicklungsendpunkt zugewiesen ist, dient auch zum Erstellen Ihres Notebook-Servers.
4. Navigieren Sie im Sicherheitsgruppenbereich zur Registerkarte Eingehend.
5. Fügen Sie eingehende Regeln hinzu, die etwa wie folgt aussehen:

Typ	Protocol (Protokoll)	Port-Bereich	Quelle
SSH	TCP	22	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

Im Folgenden ist ein Beispiel für die eingehenden Regeln der Sicherheitsgruppe aufgeführt:



Datenermittlung und Katalogisierung in AWS Glue

Das AWS Glue Data Catalog ist ein zentrales Repository, das Metadaten zu den Datensätzen Ihrer Organisation speichert. Es dient als Index für den Standort, das Schema und die Laufzeitmetriken Ihrer Datenquellen. Die Metadaten werden in Metadatentabellen gespeichert, wobei jede Tabelle einen einzelnen Datenspeicher darstellt.

Sie können den Datenkatalog mithilfe eines Crawlers auffüllen, der Ihre Datenquellen automatisch scannt und Metadaten extrahiert. Ein Crawler kann eine Verbindung zu internen (AWS basierten) und externen Datenquellen herstellen. AWS

Weitere Informationen zu den unterstützten Datenquellen finden Sie unter Unterstützte Datenquellen für das Crawlen

Sie können Tabellen im Datenkatalog auch manuell erstellen, indem Sie die Tabellenstruktur, das Schema und die Partitionierungsstruktur entsprechend Ihren spezifischen Anforderungen definieren.

Weitere Informationen zum manuellen Erstellen von Metadatentabellen finden Sie unterManuelles Definieren von Metadaten.

Sie können die Informationen im Datenkatalog verwenden, um Ihre ETL-Jobs zu erstellen und zu überwachen. Der Datenkatalog lässt sich in andere AWS Analysedienste integrieren und bietet eine einheitliche Ansicht der Datenquellen, was die Verwaltung und Analyse von Daten erleichtert.

- Amazon Athena — Speichern und Abfragen von Tabellenmetadaten im Datenkatalog für die Amazon S3 S3-Daten mithilfe von SQL.
- AWS Lake Formation — Definieren und verwalten Sie zentral detaillierte Datenzugriffsrichtlinien und überprüfen Sie den Datenzugriff.
- Amazon EMR — Greifen Sie auf Datenquellen zu, die im Datenkatalog für die Verarbeitung großer Datenmengen definiert sind.
- Amazon SageMaker AI — Schnelles und sicheres Erstellen, Trainieren und Implementieren von Modellen für maschinelles Lernen.

Hauptmerkmale des Datenkatalogs

Im Folgenden sind die wichtigsten Aspekte des Datenkatalogs aufgeführt.

Metadaten-Repository

Der Datenkatalog fungiert als zentrales Metadaten-Repository, in dem Informationen über den Speicherort, das Schema und die Eigenschaften Ihrer Datenquellen gespeichert werden. Diese Metadaten sind in Datenbanken und Tabellen organisiert, ähnlich einem herkömmlichen relationalen Datenbankkatalog.

Automatische Auffindbarkeit von Daten

AWS-Glue-Crawler s kann neue oder aktualisierte Datenquellen automatisch erkennen und katalogisieren, wodurch der Aufwand für die manuelle Metadatenverwaltung reduziert und sichergestellt wird, dass Ihr Datenkatalog erhalten bleibt up-to-date. Durch die Katalogisierung Ihrer Datenquellen erleichtert der Datenkatalog Benutzern und Anwendungen das Auffinden und Verständnis der verfügbaren Datenbestände in Ihrem Unternehmen und fördert so die Wiederverwendung von Daten und die Zusammenarbeit.

Der Datenkatalog unterstützt eine Vielzahl von Datenquellen, darunter Amazon S3, Amazon RDS, Amazon Redshift, Apache Hive und mehr. Mithilfe von s können Metadaten aus diesen Quellen automatisch abgeleitet und gespeichert werden. AWS-Glue-Crawler

Weitere Informationen finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs .

Schemaverwaltung

Der Datenkatalog erfasst und verwaltet automatisch das Schema Ihrer Datenquellen, einschließlich Schemainferenz, Weiterentwicklung und Versionierung. Sie können Ihr Schema und Ihre Partitionen im Datenkatalog mithilfe von AWS Glue ETL-Jobs aktualisieren.

Tabellenoptimierung

Um die Leseleistung von AWS Analysediensten wie Amazon Athena und Amazon EMR sowie AWS Glue ETL-Jobs zu verbessern, bietet der Datenkatalog eine verwaltete Komprimierung (ein Prozess, der kleine Amazon S3 S3-Objekte zu größeren Objekten komprimiert) für Eisberg-Tabellen im Datenkatalog. Sie können die AWS Glue Konsole, AWS Lake Formation Konsole oder AWS API verwenden, um die AWS CLI Komprimierung für einzelne Iceberg-Tabellen, die sich im Datenkatalog befinden, zu aktivieren oder zu deaktivieren.

Weitere Informationen finden Sie unter Optimieren von Iceberg-Tabellen.

Spaltenstatistiken

Sie können Statistiken auf Spaltenebene für Datenkatalogtabellen in Datenformaten wie Parquet, ORC, JSON, ION, CSV und XML berechnen, ohne zusätzliche Datenpipelines einrichten zu müssen. Spaltenstatistiken helfen Ihnen dabei, Datenprofile zu verstehen, indem sie Einblicke in die Werte innerhalb einer Spalte gewinnen. Der Datenkatalog unterstützt die Generierung von Statistiken für Spaltenwerte wie Minimalwert, Maximalwert, Gesamt-Nullwerte, Gesamtzahl unterschiedlicher Werte, durchschnittliche Länge von Werten und Gesamtzahl der Vorkommen von wahren Werten.

Weitere Informationen finden Sie unter Optimieren der Abfrageleistung mithilfe von Spaltenstatistiken.

Herkunft der Daten

Der Datenkatalog zeichnet die Transformationen und Operationen auf, die an Ihren Daten durchgeführt wurden, und stellt Informationen zur Datenherkunft bereit. Diese Informationen zur Herkunft sind für die Prüfung, Einhaltung der Vorschriften und für das Verständnis der Herkunft der Daten von Nutzen.

Integration mit anderen Diensten AWS

Der Datenkatalog lässt sich nahtlos in andere AWS Dienste wie Amazon Athena AWS Lake Formation, Amazon Redshift Spectrum und Amazon EMR integrieren. Diese Integration ermöglicht es Ihnen, Daten aus verschiedenen Datenspeichern mithilfe einer einzigen, konsistenten Metadatenebene abzufragen und zu analysieren.

Sicherheit mit Zugriffskontrolle

AWS Glue lässt sich integrieren AWS Lake Formation , um eine differenzierte Zugriffskontrolle für Datenkatalogressourcen zu unterstützen, sodass Sie Berechtigungen verwalten und den Zugriff auf Ihre Datenbestände auf der Grundlage der Richtlinien und Anforderungen Ihres Unternehmens sichern können. AWS Glue lässt sich in AWS Key Management Service (AWS KMS) integrieren, um Metadaten zu verschlüsseln, die im Datenkatalog gespeichert sind.

Themen

- Auffüllen des AWS Glue Datenkatalogs
- Auffüllen und Verwalten von Transaktionstabellen
- Verwaltung des Datenkatalogs

- Zugriff auf den Datenkatalog
- AWS Glue Bewährte Methoden für den Datenkatalog
- Überwachung der Nutzungsmetriken des Datenkatalogs in Amazon CloudWatch
- AWS Glue Schema Registry

Auffüllen des AWS Glue Datenkatalogs

Sie können den AWS Glue Data Catalog mit den folgenden Methoden auffüllen:

- AWS-Glue-Crawler — An AWS-Glue-Crawler kann Datenquellen wie Datenbanken, Data Lakes und Streaming-Daten automatisch erkennen und katalogisieren. Die Crawler sind die gängigste und empfohlene Methode zum Füllen des Datenkatalogs, da sie automatisch Metadaten für eine Vielzahl von Datenquellen erkennen und daraus ableiten können.
- Manuelles Hinzufügen von Metadaten — Sie können Datenbanken, Tabellen und Verbindungsdetails manuell definieren und sie mithilfe der Konsole, der Lake Formation AWS Glue Formation-Konsole AWS CLI, oder dem Datenkatalog hinzufügen AWS Glue APIs. Die manuelle Eingabe ist nützlich, wenn Sie Datenquellen katalogisieren möchten, die nicht gecrawlt werden können.
- Integration mit anderen AWS Diensten — Sie können den Datenkatalog mit Metadaten von Diensten wie Amazon AWS Lake Formation Athena füllen. Diese Dienste können Datenquellen im Datenkatalog erkennen und registrieren.
- Daten aus einem vorhandenen Metadaten-Repository auffüllen — Wenn Sie bereits über einen Metadatenspeicher wie Apache Hive Metastore verfügen, können AWS Glue Sie diese Metadaten in den Datenkatalog importieren. Weitere Informationen finden Sie unter Migration zwischen dem Hive Metastore und dem Hive Metastore. AWS Glue Data Catalog GitHub

Themen

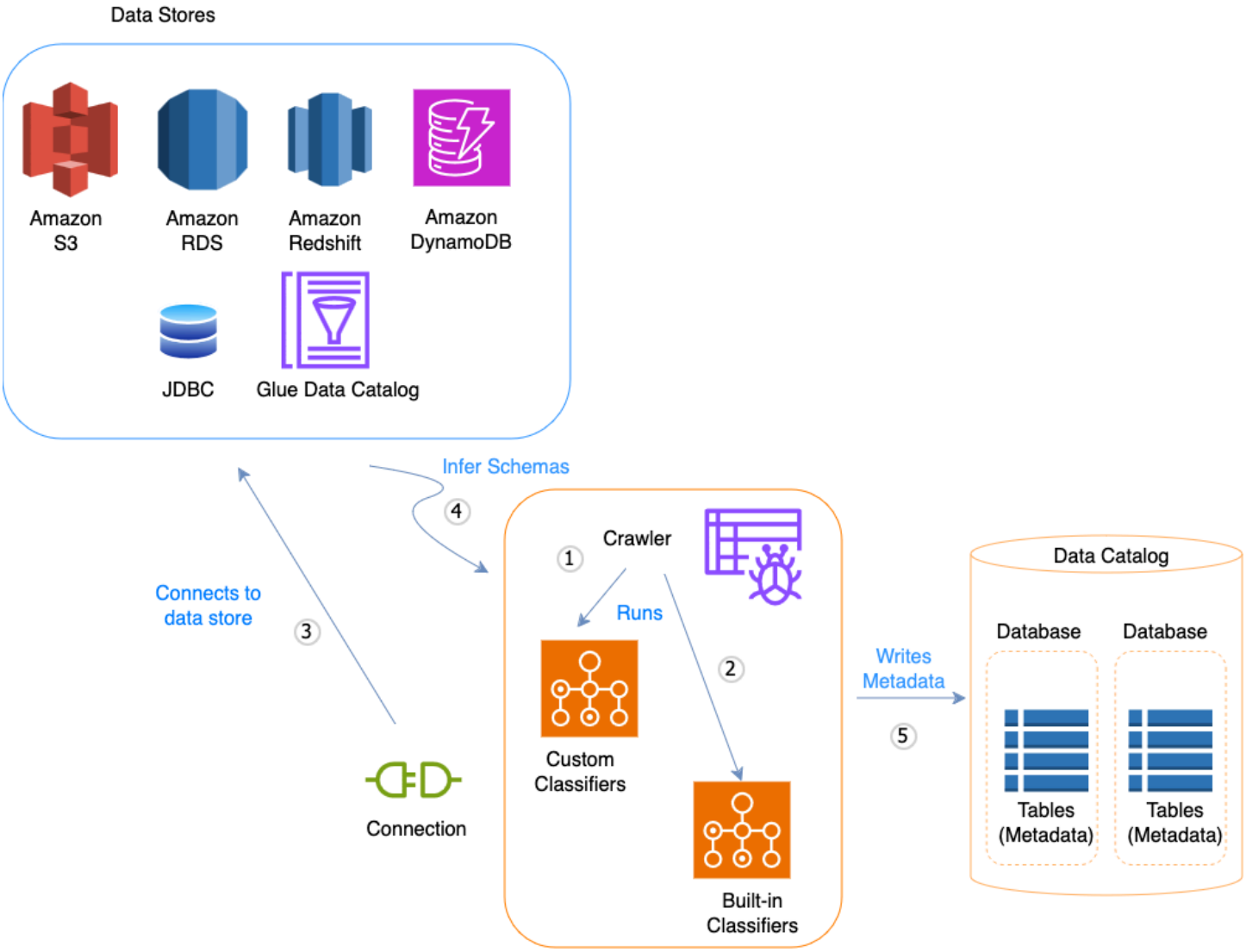
- Verwenden von Crawlern zum Auffüllen des Datenkatalogs
- Manuelles Definieren von Metadaten
- Integration mit anderen AWS Diensten
- Einstellungen für den Datenkatalog

Verwenden von Crawlern zum Auffüllen des Datenkatalogs

Sie können eine verwenden AWS-Glue-Crawler , um sie AWS Glue Data Catalog mit Datenbanken und Tabellen zu füllen. Dies ist die primäre Methode, die von den meisten AWS Glue Benutzern verwendet wird. Ein Crawler kann in einem einzigen Lauf mehrere Datenspeicher durchsuchen. Nach dem Abschluss erstellt oder aktualisiert der Crawler eine oder mehrere Tabellen in Ihrem Data Catalog. Extrahieren, Transformieren und Laden (ETL) Jobs, die Sie definieren in AWS Glue verwenden Sie diese Datenkatalogtabellen als Quellen und Ziele. Der ETL-Auftrag führt Lese- und Schreibvorgänge in Datenspeichern durch, die in den Data-Catalog-Quell- und -Zieltabellen angegeben werden.

Workflow

Das folgende Workflow-Diagramm zeigt, wie AWS Glue Crawler interagieren mit Datenspeichern und anderen Elementen, um den Datenkatalog zu füllen.



Im Folgenden wird der grundlegende Workflow beschrieben, mit dem ein Crawler den AWS Glue Data Catalog füllt:

1. Ein Crawler führt alle von Ihnen ausgewählten angepassten Classifier aus, um das Schema Ihrer Daten abzuleiten. Sie stellen den Code für angepasste Classifier bereit, die in der von Ihnen angegebenen Reihenfolge ausgeführt werden.

   Der erste angepasste Classifier, der die Struktur Ihrer Daten erfolgreich erkennt, wird zur Erstellung eines Schemas verwendet. Angepasste Classifier, die in der Liste weiter unten stehen, werden übersprungen.

2. Wenn kein angepasster Classifier mit dem Schema Ihrer Daten übereinstimmt, versuchen die integrierten Classifier, das Schema Ihrer Daten zu erkennen. Ein Beispiel für einen integrierten Classifier ist einer, der JSON erkennt.

3. Der Crawler verbindet sich mit dem Datenspeicher. Einige Datenspeicher benötigen Verbindungseigenschaften für den Crawler-Zugriff.
4. Das abgeleitete Schema wird für Ihre Daten angelegt.
5. Der Crawler schreibt Metadaten in den Data Catalog. Eine Tabellendefinition enthält Metadaten zu den Daten in Ihrem Datenspeicher. Die Tabelle wird in eine Datenbank geschrieben, die einen Tabellen-Container im Data Catalog darstellt. Attribute einer Tabelle beinhalten die Klassifizierung – d. h. ein Label, das von dem Classifier erstellt wurde, der das Tabellenschema hergeleitet hat.

Themen

- Funktionsweise von Crawlern
- Wie bestimmt ein Crawler, wann Partitionen zu erstellen sind?
- Unterstützte Datenquellen für das Crawlen
- Voraussetzungen für Crawler
- Klassifikatoren definieren und verwalten
- Konfiguration eines Crawlers
- Planen eines Crawlers
- Anzeigen von Crawler-Ergebnissen und -Details
- Crawler-Verhalten anpassen
- Tutorial: Hinzufügen eines AWS Glue Crawler

Funktionsweise von Crawlern

Wenn ein Crawler ausgeführt wird, werden für das Abfragen eines Datenspeichers folgende Aktionen verwendet:

- Klassifizieren von Daten, um das Format, das Schema und verknüpfte Eigenschaften der Rohdaten zu bestimmen – Sie können die Ergebnisse der Klassifizierung konfigurieren, indem Sie einen benutzerdefinierten Classifier erstellen.
- Gruppieren von Daten in Tabellen oder Partitionen – Daten werden basierend auf der Crawler-Heuristik gruppiert.
- Schreiben von Metadaten in den Data Catalog – Sie können konfigurieren, wie der Crawler Tabellen und Partitionen hinzufügt, aktualisiert und löscht.

Beim Definieren eines Crawlers wählen Sie einen oder mehrere Classifier aus, die das Format Ihrer Daten bewerten, um ein Schema abzuleiten. Wenn der Crawler ausgeführt wird, wird der erste Classifier in der Liste, der erfolgreich Ihren Datenspeicher erkennt, zum Erstellen eines Schemas für Ihre Tabelle verwendet. Sie können integrierte Classifier verwenden oder eigene Classifier definieren. Sie definieren Ihre benutzerdefinierten Klassifikatoren in einem separaten Vorgang, bevor Sie die Crawler definieren. AWS Glue bietet integrierte Klassifikatoren zum Ableiten von Schemas aus gängigen Dateien mit Formaten wie JSON, CSV und Apache Avro. Die aktuelle Liste der integrierten Klassifikatoren finden Sie in AWS Glue, finden Sie unter Integrierte Klassifikatoren.

Die Metadatentabellen, die von einem Crawler erstellt werden, sind in einer Datenbank enthalten, wenn Sie einen Crawler definieren. Wenn Ihr Crawler keine Datenbank definiert, werden Ihre Tabellen in der Standarddatenbank abgelegt. Darüber hinaus verfügt jede Tabelle über eine Klassifizierungsspalte, die vom Classifier gefüllt wird, der den Datenspeicher als erster erfolgreich erkennt.

Wenn die Datei, die durchsucht wird, komprimiert ist, muss der Crawler sie herunterladen, um sie zu verarbeiten. Wenn ein Crawler ausgeführt wird, fragt er Daten ab, um deren Format und Komprimierungstyp zu bestimmen, und schreibt diese Eigenschaften in den Data Catalog. Einige Dateiformate (z. B. Apache Parquet) ermöglichen das Komprimieren von Teilen der Datei beim Schreiben. Bei diesen Dateien sind die komprimierten Daten ein interner Bestandteil der Datei, und AWS Glue füllt die `compressionType` Eigenschaft nicht auf, wenn Tabellen in den Datenkatalog geschrieben werden. Wenn dagegen eine gesamte Datei durch einen Komprimierungsalgorithmus komprimiert wird (z. B. gzip), wird die `compressionType`-Eigenschaft gefüllt, wenn die Tabellen in den Data Catalog geschrieben werden.

Der Crawler generiert die Namen für die Tabellen, die er erstellt. Die Namen der Tabellen, die in der gespeichert sind, AWS Glue Data Catalog folgen diesen Regeln:

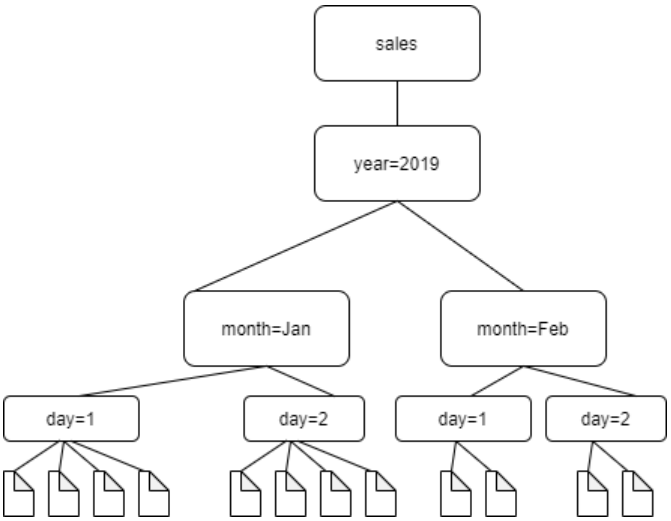
- Nur alphanumerische Zeichen und Unterstriche (_) sind erlaubt.
- Ein benutzerdefinierter Präfix darf nicht länger sein als 64 Zeichen.
- Die maximale Länge des Namens darf nicht mehr als 128 Zeichen sein. Der Crawler kürzt generierte Namen, um sie an die maximale Größe anzupassen.
- Wenn doppelte Tabellennamen auftreten, fügt der Crawler dem Namen einen Suffix als Hash-Zeichenfolge hinzu.

Wenn Ihr Crawler mehr als einmal ausgeführt wird, z. B. bei einem Zeitplan, sucht er nach neuen oder geänderten Dateien oder Tabellen in Ihrem Datenspeicher. Die Ausgabe des Crawlers umfasst die neuen Tabellen und Partitionen, die seit einer vorherigen Ausführung gefunden wurden.

Wie bestimmt ein Crawler, wann Partitionen zu erstellen sind?

Wann ein AWS Glue Der Crawler scannt den Amazon S3 S3-Datenspeicher und erkennt mehrere Ordner in einem Bucket. Er bestimmt den Stamm einer Tabelle in der Ordnerstruktur und welche Ordner Partitionen einer Tabelle sind. Der Name der Tabelle basiert auf dem Amazon-S3-Präfix oder Ordnernamen. Sie stellen einen Include path (Include-Pfad) bereit, der auf die zu durchsuchende Ordnerebene zeigt. Wenn die Mehrzahl der Schemas auf einer Ordnerebene ähnlich sind, erstellt der Crawler Partitionen der Tabelle anstelle separater Tabellen. Damit der Crawler separate Tabellen erstellt, fügen Sie den Stammordner einer jeden Tabelle als separaten Datenspeicher hinzu, wenn Sie den Crawler definieren.

Sehen Sie sich beispielsweise die folgende Amazon-S3-Ordnerstruktur an:



Die Pfade zu den vier untersten Ordnern lauten wie folgt:

```
S3://sales/year=2019/month=Jan/day=1
S3://sales/year=2019/month=Jan/day=2
S3://sales/year=2019/month=Feb/day=1
S3://sales/year=2019/month=Feb/day=2
```

Gehen Sie davon aus, dass das Crawler-Ziel auf `Sales` festgelegt wird und dass alle Dateien im `day=`*n*-Ordner das gleiche Format (z. B. JSON, nicht verschlüsselt) sowie dieselben oder sehr ähnliche Schemas haben. Der Crawler erstellt eine einzelne Tabelle mit vier Partitionen, mit Partitionsschlüsseln `year`, `month` und `day`.

Sehen Sie sich im nächsten Beispiel die folgende Amazon-S3-Ordnerstruktur an:

```
s3://bucket01/folder1/table1/partition1/file.txt
s3://bucket01/folder1/table1/partition2/file.txt
s3://bucket01/folder1/table1/partition3/file.txt
s3://bucket01/folder1/table2/partition4/file.txt
s3://bucket01/folder1/table2/partition5/file.txt
```

Wenn sich die Schemas für `table1` und `table2` ähneln und im Crawler ein einzelner Datenspeicher mit Include path (Include-Pfad) `s3://bucket01/folder1/` definiert ist, erstellt der Crawler eine einzelne Tabelle mit zwei Partitionsschlüsselspalten. Die erste Partitionsschlüsselspalte enthält `table1` und `table2`. Der zweite Partitionsschlüsselspalte enthält `partition1` bis `partition3` für die `table1`-Partition sowie `partition4` und `partition5` für die `table2`-Partition. Zum Erstellen zweier separater Tabellen definieren Sie den Crawler mit zwei Datenspeichern. In diesem Beispiel definieren Sie den ersten Include path (Include-Pfad) als `s3://bucket01/folder1/table1/` und den zweiten als `s3://bucket01/folder1/table2`.

> **Note**
>
> In Amazon Athena entspricht jede Tabelle einem Amazon-S3-Präfix mit allen darin enthaltenen Objekten. Wenn Objekte unterschiedliche Schemas haben, erkennt Athena unterschiedliche Objekte innerhalb desselben Präfix nicht als separate Tabellen. Dies kann der Fall sein, wenn ein Crawler mehrere Tabellen aus demselben Amazon-S3-Präfix erstellt. Dies kann zu ergebnislosen Abfragen in Athena führen. Damit Athena Tabellen richtig erkennen und abfragen kann, erstellen Sie den Crawler mit einem separaten Include path (Include-Pfad) für die unterschiedlichen Tabellenschemas in der Amazon-S3-Ordnerstruktur. Weitere Informationen finden Sie unter Bewährte Methoden bei der Verwendung von Athena mit AWS Glueund dieser AWS Knowledge Center-Artikel.

Unterstützte Datenquellen für das Crawlen

Crawler können sowohl dateibasierte als auch tabellenbasierte Datenspeicher durchsuchen.

Zugriffstyp, den Crawler verwendet	Datastores
Native Clients	• Amazon Simple Storage Service (Amazon-S3)

Zugriffstyp, den Crawler verwendet	Datastores
	• Amazon-DynamoDB • Delta Lake 2.0.x • Apache Iceberg 1.5 • Apache Hudi 0.14
JDBC	Amazon Redshift Snowflake Innerhalb von Amazon Relational Database Service (Amazon RDS) oder außerhalb von Amazon RDS: • Amazon Aurora • MariaDB • Microsoft SQL Server • MySQL • Oracle • PostgreSQL
MongoDB-Client	• MongoDB • MongoDB Atlas • Amazon DocumentDB (mit MongoDB-Kompatibilität)

> **Note**
>
> Derzeit AWS Glue unterstützt keine Crawler für Datenströme.

Für JDBC-, MongoDB-, MongoDB Atlas- und Amazon DocumentDB DocumentDB-Datenspeicher (mit MongoDB-Kompatibilität) müssen Sie eine angeben AWS Glue Verbindung, über die der Crawler eine Verbindung zum Datenspeicher herstellen kann. Für Amazon S3 können Sie optional eine Verbindung vom Typ Netzwerk angeben. Eine Verbindung ist ein Data-Catalog-Objekt, in dem Verbindungsinformationen wie Anmeldeinformationen, URL, Informationen zu Amazon Virtual

Private Cloud und mehr gespeichert werden. Weitere Informationen finden Sie unter Herstellen einer Verbindung zu Daten.

Die folgenden Treiberversionen werden vom Crawler unterstützt:

Produkt	Der vom Crawler unterstützte Treiber
PostgreSQL	42.2.1
Amazon Aurora	Wie bei nativen Crawler-Treibern
MariaDB	8.0.13
Microsoft SQL Server	6.1.0
MySQL	8.0.13
Oracle	11.2.2
Amazon Redshift	4.1
Snowflake	3.13,20
MongoDB	4.7.2
MongoDB Atlas	4.7.2

Im Folgenden finden Sie Hinweise zu den verschiedenen Datenspeichern.

Amazon S3

Sie können einen Pfad in Ihrem Konto oder in einem anderen Konto durchsuchen. Wenn alle Amazon-S3-Dateien in einem Ordner über das gleiche Schema verfügen, erstellt der Crawler eine Tabelle. Wenn das Amazon-S3-Objekt partitioniert ist, wird nur eine Metadatentabelle erstellt und dem Data Catalog für diese Tabelle Partitionsinformationen hinzugefügt.

Amazon S3 und Amazon DynamoDB

Crawler verwenden eine AWS Identity and Access Management (IAM-) Rolle, um Zugriff auf Ihre Datenspeicher zu erhalten. Die Rolle, die Sie an den Crawler übergeben, muss über die

Berechtigung für den Zugriff auf Amazon-S3-Pfade und Amazon-DynamoDB-Tabellen verfügen, die durchsucht werden.

Amazon-DynamoDB

Bei der Definition eines Crawlers mithilfe der AWS Glue Konsole, Sie geben eine DynamoDB-Tabelle an. Wenn Sie die verwenden AWS Glue API, Sie können eine Liste von Tabellen angeben. Sie können nur eine kleine Stichprobe der Daten durchforsten, um die Laufzeiten von Crawlern zu reduzieren.

Delta Lake

Für jeden Delta-Lake-Datenspeicher geben Sie an, wie diese Delta-Tabelle erstellt werden soll:

- Erstellen nativer Tabellen: Ermöglicht die Integration mit Abfragemodulen, die die direkte Abfrage des Delta-Transaktionsprotokolls unterstützen. Weitere Informationen finden Sie unter Abfragen von Delta-Lake-Tabellen.
- Erstellen von Symlink-Tabellen: Erstellen Sie auf der Grundlage der angegebenen Konfigurationsparameter einen `_symlink_manifest`-Ordner mit Manifestdateien, die durch die Partitionsschlüssel partitioniert sind.

Iceberg

Sie geben für jeden Iceberg-Datenspeicher einen Amazon-S3-Pfad an, der die Metadaten für Ihre Iceberg-Tabellen enthält. Wenn der Crawler die Metadaten der Iceberg-Tabelle entdeckt, registriert er sie im Data Catalog. Sie können einen Zeitplan für den Crawler festlegen, um die Tabellen auf dem neuesten Stand zu halten.

Sie können diese Parameter für den Datenspeicher definieren:

- Ausschlüsse: Ermöglicht das Überspringen bestimmter Ordner.
- Maximale Durchquerungstiefe: Legt die Tiefenbeschränkung fest, die der Crawler in Ihrem Amazon-S3-Bucket crawlen kann. Die standardmäßige maximale Durchquerungstiefe beträgt 10 und die maximal einstellbare Tiefe beträgt 20.

Hudi

Geben Sie für jeden Hudi-Datenspeicher einen Amazon-S3-Pfad an, der die Metadaten für Ihre Hudi-Tabellen enthält. Wenn der Crawler die Metadaten der Hudi-Tabelle entdeckt, registriert er sie im Data Catalog. Sie können einen Zeitplan für den Crawler festlegen, um die Tabellen auf dem neuesten Stand zu halten.

Sie können diese Parameter für den Datenspeicher definieren:

- Ausschlüsse: Ermöglicht das Überspringen bestimmter Ordner.
- Maximale Durchquerungstiefe: Legt die Tiefenbeschränkung fest, die der Crawler in Ihrem Amazon-S3-Bucket crawlen kann. Die standardmäßige maximale Durchquerungstiefe beträgt 10 und die maximal einstellbare Tiefe beträgt 20.

> **Note**
>
> Zeitstempelspalten mit `millis` als logischen Typen werden aufgrund einer Inkompatibilität mit Hudi 0.13.1 und Zeitstempeltypen als `bigint` interpretiert. Eine Lösung wird möglicherweise in der kommenden Hudi-Version bereitgestellt.

Hudi-Tabellen werden wie folgt kategorisiert, mit jeweils spezifischen Auswirkungen:

- Copy on Write (CoW, Beim Schreiben kopieren): Daten werden in einem spaltenbasierten Format (Parquet) gespeichert, und jedes Update erstellt während eines Schreibvorgangs eine neue Version von Dateien.
- Merge on Read (MoR, Beim Lesen zusammenführen): Daten werden mit einer Kombination aus spalten- (Parquet) und zeilenbasierten (Avro) Formaten gespeichert. Updates werden in zeilenbasierten Delta-Dateien protokolliert und nach Bedarf komprimiert, um neue Versionen der Spaltendateien zu erstellen.

Bei CoW-Datasets wird jedes Mal, wenn ein Datensatz aktualisiert wird, die Datei, die den Datensatz enthält, mit den aktualisierten Werten neu geschrieben. Bei einem MoR-Datensatz schreibt Hudi jedes Mal, wenn es eine Aktualisierung gibt, nur die Zeile für den geänderten Datensatz. MoR eignet sich besser für schreib- oder änderungsintensive Workloads mit weniger Lesevorgängen. CoW eignet sich besser für leseintensive Workloads für Daten, die sich seltener ändern.

Hudi bietet drei Abfragetypen für den Zugriff auf die Daten:

- Snapshot-Abfragen: Abfragen, die den neuesten Snapshot der Tabelle ab einer bestimmten Commit- oder Komprimierungsaktion anzeigen. Bei MoR-Tabellen stellen Snapshot-Abfragen den neuesten Status der Tabelle dar, indem die Basis- und Deltadateien des letzten Datei-Slices zum Zeitpunkt der Abfrage zusammengeführt werden.
- Inkrementelle Abfragen: Bei Abfragen werden nur neue Daten angezeigt, die seit einem bestimmten Commit/einer bestimmten Komprimierung in die Tabelle geschrieben wurden. Dies bietet effektiv Änderungsströme, um inkrementelle Data-Pipelines zu ermöglichen.

- Lesen von optimierten Abfragen: Bei MoR-Tabellen zeigen Abfragen die neuesten komprimierten Daten an. Bei CoW-Tabellen sehen Abfragen die neuesten festgeschriebenen Daten.

Bei Copy-On-Write Tabellen erstellen die Crawler mit der ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat` Serde eine einzelne Tabelle im Datenkatalog.

Bei Merge-On-Read Tabellen erstellt der Crawler zwei Tabellen im Datenkatalog für dieselbe Tabellenposition:

- Eine Tabelle mit Suffix`_ro`, die das Serde verwendet. ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat`
- Eine Tabelle mit Suffix`_rt`, die die RealTime Serde verwendet und Snapshot-Abfragen ermöglicht:. `org.apache.hudi.hadoop.realtime.HoodieParquetRealtimeInputFormat`

MongoDB und Amazon DocumentDB (mit MongoDB-Kompatibilität)

MongoDB-Versionen ab Version 3.2 werden unterstützt. Sie können nur eine kleine Stichprobe der Daten durchforsten, um die Laufzeiten von Crawlern zu reduzieren.

Relationale Datenbank

Die Authentifizierung erfolgt mit einem Datenbankbenutzernamen und einem -Passwort. Abhängig vom Typ der Datenbank-Engine können Sie auswählen, welche Objekte durchsucht werden, beispielsweise Datenbanken, Schemas oder Tabellen.

Snowflake

Der Snowflake JDBC-Crawler unterstützt das Crawling der Tabelle, der externen Tabelle, der Ansicht und der materialisierten Ansicht. Die materialisierte Ansichtsdefinition wird nicht ausgefüllt.

Bei externen Snowflake-Tabellen durchsucht der Crawler nur, wenn er auf einen Amazon-S3-Speicherort verweist. Zusätzlich zum Tabellenschema durchsucht der Crawler auch den Amazon-S3-Speicherort, das Dateiformat und die Ausgabe als Tabellenparameter in der Data-Catalog-Tabelle. Beachten Sie, dass die Partitionsinformationen der partitionierten externen Tabelle nicht ausgefüllt werden.

ETL wird derzeit nicht für Data-Catalog-Tabellen unterstützt, die mit dem Snowflake-Crawler erstellt wurden.

Voraussetzungen für Crawler

Der Crawler übernimmt die Berechtigungen der AWS Identity and Access Management (IAM-) Rolle, die Sie bei der Definition angeben. Diese IAM-Rolle muss über Berechtigungen zum Extrahieren von Daten aus Ihrem Datenspeicher und zum Schreiben in den Data Catalog verfügen. Die AWS Glue-Konsole führt nur IAM-Rollen auf, denen eine Vertrauensrichtlinie für den AWS Glue-Prinzipal-Service angefügt ist. Von der Konsole aus können Sie auch eine IAM-Rolle mit einer IAM-Richtlinie für den Zugriff auf Amazon-S3-Datenspeicher erstellen, auf die der Crawler zugreift. Weitere Informationen zum Bereitstellen von Rollen für AWS Glue finden Sie unter Identitätsbasierte Richtlinien für Glue AWS.

> **Note**
>
> Beim Crawlen eines Delta Lake-Datenspeichers benötigen Sie Read/Write Berechtigungen für den Amazon S3 S3-Standort.

Für Ihren Crawler können Sie eine Rolle erstellen und die folgenden Richtlinien anfügen:

- Die `AWSGlueServiceRole` AWS verwaltete Richtlinie, die die erforderlichen Berechtigungen für den Datenkatalog gewährt
- Eine Inline-Richtlinie, die Berechtigungen für die Datenquelle erteilt.
- Eine Inline-Richtlinie, die `iam:PassRole` Berechtigungen für die Rolle gewährt.

Ein schnellerer Ansatz besteht darin, den Crawler-Assistenten der AWS Glue-Konsole eine Rolle für Sie erstellen zu lassen. Die Rolle, die sie erstellt, ist speziell für den Crawler bestimmt und umfasst die `AWSGlueServiceRole` AWS verwaltete Richtlinie sowie die erforderliche Inline-Richtlinie für die angegebene Datenquelle.

Wenn Sie eine vorhandene Rolle für einen Crawler angeben, stellen Sie sicher, dass dieser die `AWSGlueServiceRole`-Richtlinie oder eine gleichwertige Version dieser Richtlinie (oder eine abgespeckte Version) sowie die erforderlichen Inline-Richtlinien enthält. Für einen Amazon-S3-Datenspeicher wäre die Inline-Richtlinie beispielsweise mindestens die folgende:

JSON

```
{
```

```
   "Version": "2012-10-17",
    "Statement": [
        {
          "Effect": "Allow",
          "Action": [
              "s3:GetObject"
          ],
          "Resource": [
              "arn:aws:s3:::bucket/object*"
          ]
        }
    ]
}
```

Für einen Amazon-DynamoDB-Datenspeicher wäre die Richtlinie beispielsweise mindestens die folgende:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:region:account-id:table/table-name*"
      ]
    }
  ]
}
```

Wenn der Crawler verschlüsselte Amazon S3 S3-Daten liest AWS Key Management Service (AWS KMS), muss die IAM-Rolle außerdem über die Entschlüsselungsberechtigung für den Schlüssel verfügen. AWS KMS Weitere Informationen finden Sie unter Schritt 2: Erstellen einer IAM-Rolle für AWS Glue.

Klassifikatoren definieren und verwalten

Ein Classifier liest die Daten in einem Datenspeicher. Wenn er das Format der Daten erkennt, erzeugt er ein Schema. Der Classifier gibt auch eine Zahl für die Gewissheit zurück, um anzugeben, wie sicher die Formaterkennung war.

AWS Glue bietet eine Reihe integrierter Klassifikatoren, aber Sie können auch benutzerdefinierte Klassifikatoren erstellen. AWS Glue ruft zuerst benutzerdefinierte Klassifikatoren in der Reihenfolge auf, die Sie in Ihrer Crawler-Definition angegeben haben. Abhängig von den Ergebnissen, die von benutzerdefinierten Klassifikatoren zurückgegeben werden, AWS Glue kann auch integrierte Klassifikatoren aufrufen. Wenn ein Klassifikator `certainty=1.0` während der Verarbeitung zurückkehrt, bedeutet dies, dass er zu 100 Prozent sicher ist, dass er das richtige Schema erstellen kann. AWS Glue verwendet dann die Ausgabe dieses Klassifikators.

Wenn kein Klassifikator zurückkehrt, `certainty=1.0` AWS Glue verwendet die Ausgabe des Klassifikators mit der höchsten Sicherheit. Wenn kein Klassifikator eine Sicherheit zurückgibt, die größer ist als, `0.0` AWS Glue gibt die Standardklassifizierungszeichenfolge von zurück. `UNKNOWN`

Wann verwende ich einen Classifier?

Sie verwenden Classifier, wenn Sie einen Datenspeicher durchsuchen, um Metadatentabellen in AWS Glue Data Catalog zu definieren. Sie können Ihren Crawler mit einer geordneten Gruppe von Classifiern einrichten. Wenn der Crawler einen Classifier aufruft, bestimmt dieser, ob die Daten erkannt werden. Wenn der Classifier die Daten nicht erkennen kann oder nicht 100 %ig sicher ist, ruft der Crawler den nächsten Classifier in der Liste auf, um festzustellen ob dieser die Daten erkennen kann.

Weitere Informationen zum Erstellen eines Klassifikators mit dem AWS Glue Konsole finden Sie unterErstellen von Klassifikatoren mit dem AWS Glue Konsole.

Benutzerdefinierte Classifier

Die Ausgabe eines Classifiers enthält eine Zeichenfolge, die die Klassifizierung oder das Format der Datei (z. B. `json`) sowie das Schema der Datei angibt. Bei einem benutzerdefinierten Classifier definieren Sie die Logik für die Erstellung des Schemas basierend auf dem Classifier-Typ. Classifier-Typen umfassen das Festlegen von Schemas basierend auf Grok-Mustern, XML-Tags und JSON-Pfaden.

Wenn Sie eine Classifier-Definition ändern, werden Daten, die zuvor mit dem Crawler durchsucht wurden, nicht neu klassifiziert. Ein Crawler führt eine Nachverfolgung zuvor durchsuchter Daten

durch. Neue Daten werden mit dem aktualisierten Classifier klassifiziert. Dies kann ein aktualisiertes Schema zur Folge haben. Hat sich das Schema Ihrer Daten weiterentwickelt, aktualisieren Sie den Classifier, um allen Schemaänderungen Rechnung zu tragen, wenn Ihr Crawler ausgeführt wird. Zur Neuklassifizierung von Daten zur Korrektur eines fehlerhaften Classifiers erstellen Sie einen neuen Crawler mit dem aktualisierten Classifier.

Weitere Informationen zum Erstellen benutzerdefinierter Klassifikatoren finden Sie in AWS Glue, finden Sie unter Schreiben von benutzerdefinierten Klassifikatoren für verschiedene Datenformate.

> **Note**
>
> Wenn Ihr Datenformat von einem der integrierten Classifier erkannt wird, müssen Sie keinen benutzerdefinierten Classifier erstellen.

Integrierte Klassifikatoren

AWS Glue bietet integrierte Klassifikatoren für verschiedene Formate, darunter JSON, CSV, Webprotokolle und viele Datenbanksysteme.

Wenn AWS Glue findet keinen benutzerdefinierten Klassifikator, der mit hundertprozentiger Sicherheit zum Eingabedatenformat passt, ruft die integrierten Klassifikatoren in der in der folgenden Tabelle angegebenen Reihenfolge auf. Die integrierten Classifier geben ein Ergebnis zurück, um anzugeben, ob das Format übereinstimmt (`certainty=1.0`) oder nicht übereinstimmt (`certainty=0.0`). Der erste Classifier mit `certainty=1.0` stellt die Klassifizierungszeichenfolge und das Schema für eine Metadatentabelle in Ihrem Data Catalog bereit.

Classifier-Typ	Klassifizierungszeichenfolge	Hinweise
Apache Avro	`avro`	Liest das Schema am Anfang der Datei, um das Format zu bestimmen.
Apache ORC	`orc`	Liest die Metadaten der Datei zu Bestimmung des Formats.
Apache Parquet	`parquet`	Liest das Schema am Ende der Datei, um das Format zu bestimmen.

Classifier-Typ	Klassifizierungszeichenfolge	Hinweise
JSON	`json`	Liest den Anfang der Datei, um das Format zu bestimmen.
Binäres JSON	`bson`	Liest den Anfang der Datei, um das Format zu bestimmen.
XML	`xml`	Liest den Anfang der Datei, um das Format zu bestimmen. AWS Glue bestimmt das Tabellenschema auf der Grundlage von XML-Tags im Dokument. Weitere Informationen zum Erstellen eines benutzerdefinierten XML-Classifiers zur Angabe von Zeilen in dem Dokument finden Sie unter Angepasste XML-Classifier schreiben.
Amazon Ion	`ion`	Liest den Anfang der Datei, um das Format zu bestimmen.
Kombiniertes Apache-Protokoll	`combined_apache`	Legt Protokollformate über ein Grok-Muster fest.
Apache-Protokoll	`apache`	Legt Protokollformate über ein Grok-Muster fest.
Linux-Kernel-Protokoll	`linux_kernel`	Legt Protokollformate über ein Grok-Muster fest.
Microsoft-Protokoll	`microsoft_log`	Legt Protokollformate über ein Grok-Muster fest.
Ruby-Protokoll	`ruby_logger`	Liest den Anfang der Datei, um das Format zu bestimmen.
Squid 3.x-Protokoll	`squid`	Liest den Anfang der Datei, um das Format zu bestimmen.

Classifier-Typ	Klassifizierungsze ichenfolge	Hinweise
Redis-Übe rwachungsprotokoll	`redismonlog`	Liest den Anfang der Datei, um das Format zu bestimmen.
Redis-Protokoll	`redislog`	Liest den Anfang der Datei, um das Format zu bestimmen.
CSV	`csv`	Prüft auf die folgenden Trennzeichen: Komma (,), Verkettungszeichen (|), Tabulator (\t), Semikolon (;) und Strg-A (\u0001). Strg-A ist das Unicode-S teuerzeichen für `Start Of Heading`.
Amazon Redshift	`redshift`	Verwendet eine JDBC-Verbindung zum Importieren von Metadaten.
MySQL	`mysql`	Verwendet eine JDBC-Verbindung zum Importieren von Metadaten.
PostgreSQL	`postgresql`	Verwendet eine JDBC-Verbindung zum Importieren von Metadaten.
Oracle-Datenbank	`oracle`	Verwendet eine JDBC-Verbindung zum Importieren von Metadaten.
Microsoft SQL Server	`sqlserver`	Verwendet eine JDBC-Verbindung zum Importieren von Metadaten.
Amazon-Dy namoDB	`dynamodb`	Liest Daten aus der DynamoDB-Tabelle.

Dateien in den folgenden komprimierten Formaten können klassifiziert werden:

- ZIP (unterstützt für Archive, die nur eine einzige Datei enthalten). Beachten Sie, dass Zip in anderen Services nicht gut unterstützt wird (wegen des Archivs).
- BZIP
- GZIP

- LZ4
- Snappy (unterstützt sowohl für Standard- und Hadoop-native Snappy-Formate)

Integrierter CSV-Classifier

Der integrierte CSV-Klassifikator analysiert den Inhalt der CSV-Datei, um das Schema für eine AWS Glue Tabelle. Dieser Classifier prüft auf folgende Trennzeichen:

- Komma (,)
- Pipe (|)
- Tab (\t)
- Semikolon (;)
- Strg-A (\ u0001)

  Strg-A ist das Unicode-Steuerzeichen für `Start Of Heading`.

Damit die Tabelle als CSV klassifiziert wird, muss das Tabellenschema mindestens zwei Spalten und zwei Datenzeilen aufweisen. Der CSV-Classifier nutzt eine Reihe von Heuristiken, um zu ermitteln, ob ein Header in einer bestimmten Datei vorhanden ist. Kann der Classifier keinen Header in der ersten Datenzeile feststellen, werden Spaltenüberschriften als `col1`, `col2` `col3` und so weiter angezeigt. Der integrierte CSV-Classifier bestimmt, ob ein Header durch Auswertung der folgenden Merkmale der Datei abgeleitet wird:

- Jede Spalte in einem potenziellen Header analysiert einen STRING-Datentyp.
- Mit Ausnahme der letzten Spalte hat jede Spalte in einem potenziellen Header einen Inhalt mit weniger als 150 Zeichen. Damit ein abschließendes Trennzeichen möglich ist, kann die letzte Spalte in der gesamten Datei leer sein.
- Jede Spalte in einer potenziellen Kopfzeile muss dem entsprechen AWS Glue `regex`Anforderungen für einen Spaltennamen.
- Die Kopfzeile muss sich ausreichend von den anderen Datenzeilen unterscheiden. Zu diesem Zweck muss mindestens eine Zeile nicht als STRING-Type analysiert werden. Wenn alle Spalten vom Typ STRING sind, unterscheidet sich die erste Datenzeile nicht ausreichend genug von den folgenden Zeilen, um als Header verwendet werden zu können.

> **Note**
>
> Wenn der integrierte CSV-Klassifikator Ihren nicht erstellt AWS Glue Wie Sie möchten, können Sie möglicherweise eine der folgenden Alternativen verwenden:
>
> - Ändern Sie die Spaltennamen im Data Catalog, setzen Sie `SchemaChangePolicy` auf LOG und legen Sie die Partitionsausgabekonfiguration für zukünftige Crawler-Ausführungen auf `InheritFromTable` fest.
> - Erstellen Sie einen benutzerdefinierten Grok-Classifier, um die Daten zu analysieren und die von Ihnen gewünschten Spalten zuzuweisen.
> - Der integrierte CSV-Classifier erstellt Tabellen, die auf `LazySimpleSerDe` als Serialisierungsbibliothek verweisen. Dies ist eine gute Möglichkeit für die Typinferenz. Wenn die CSV-Daten jedoch Zeichenketten in Anführungszeichen enthalten, bearbeiten Sie die Tabellendefinition und ändern Sie die SerDe Bibliothek in`OpenCSVSerDe`. Passen Sie alle abgeleiteten Typen an STRING an, setzen Sie `SchemaChangePolicy` auf LOG und legen Sie die Partitionsausgabekonfiguration für zukünftige Crawler-Ausführungen auf `InheritFromTable` fest. Weitere Informationen zu SerDe Bibliotheken finden Sie unter SerDe Referenz im Amazon Athena Athena-Benutzerhandbuch.

Schreiben von benutzerdefinierten Klassifikatoren für verschiedene Datenformate

Sie können einen angepassten Classifier bereitstellen, um Ihre Daten in AWS Glue zu klassifizieren. Sie können einen benutzerdefinierten Klassifikator mithilfe eines Grok-Musters, eines XML-Tags, JavaScript Object Notation (JSON) oder kommagetrennter Werte (CSV) erstellen. Importieren in &S3; AWS Glue Der Crawler ruft einen benutzerdefinierten Klassifikator auf. Wenn der Classifier die Daten erkennt, gibt er die Klassifizierung und das Schema der Daten an den Crawler zurück. Möglicherweise müssen Sie einen angepassten Classifier definieren, wenn Ihre Daten nicht mit den integrierten Classifiern übereinstimmen oder wenn Sie die vom Crawler erstellten Tabellen anpassen möchten.

Weitere Informationen zum Erstellen eines Klassifikators mit dem AWS Glue Konsole finden Sie unterErstellen von Klassifikatoren mit dem AWS Glue Konsole.

AWS Glue führt benutzerdefinierte Klassifikatoren vor integrierten Klassifikatoren in der von Ihnen angegebenen Reihenfolge aus. Wenn ein Crawler einen Classifier findet, der zu den Daten passt, werden die Klassifizierungszeichenkette und das Schema für die Definition von Tabellen verwendet, die in Ihren AWS Glue Data Catalog geschrieben werden.

Themen

- Angepasste Grok-Classifier schreiben
- Angepasste XML-Classifier schreiben
- Angepasste JSON-Classifier schreiben
- Angepasste CSV-Classifier schreiben

Angepasste Grok-Classifier schreiben

Grok ist ein Werkzeug, das verwendet wird, um Textdaten über ein passendes Muster zu analysieren. Ein Grok-Muster ist ein benannter Satz regulärer Ausdrücke (Regex), die verwendet werden, um Daten zeilenweise abzugleichen. AWS Glue verwendet Grok-Muster, um das Schema Ihrer Daten abzuleiten. Wenn ein Grok-Muster mit Ihren Daten übereinstimmt, AWS Glue verwendet das Muster, um die Struktur Ihrer Daten zu bestimmen und sie Feldern zuzuordnen.

AWS Glue bietet viele integrierte Muster, oder Sie können Ihre eigenen definieren. Sie können ein Grok-Muster mit Hilfe von integrierten Mustern und angepassten Mustern in Ihrer angepassten Classifier-Definition erstellen. Sie können ein Grok-Muster anpassen, um spezielle Textdateiformate zu klassifizieren.

> **Note**
>
> AWS Glue Benutzerdefinierte Grok-Klassifikatoren verwenden die `GrokSerDe` Serialisierungsbibliothek für Tabellen, die in der erstellt wurden. AWS Glue Data Catalog Wenn Sie das AWS Glue Data Catalog mit Amazon Athena, Amazon EMR oder Redshift Spectrum verwenden, finden Sie in der Dokumentation zu diesen Diensten Informationen zur Unterstützung von. `GrokSerDe` Derzeit treten möglicherweise Probleme mit der Abfrage von Tabellen auf, die mit `GrokSerDe` von Amazon EMR und Redshift Spectrum erstellt wurden.

Im Folgenden wird die grundlegende Syntax für die Komponenten eines Grok-Musters beschrieben:

```
%{PATTERN:field-name}
```

Daten, die mit dem benannten `PATTERN` übereinstimmen, werden der `field-name`-Spalte im Schema zugeordnet. Sie erhalten den Standarddatentyp `string`. Optional kann der Datentyp für das Feld als `byte`, `boolean`, `double`, `short`, `int`, `long` oder `float` in das resultierende Schema übernommen werden.

```
%{PATTERN:field-name:data-type}
```

Um z. B. ein num-Feld auf einen `int`-Datentyp festzulegen, können Sie das folgende Muster verwenden:

```
%{NUMBER:num:int}
```

Muster können aus anderen Mustern zusammengesetzt werden. Sie können z. B. ein Muster für einen SYSLOG-Zeitstempel verwenden, das durch Muster für Monat, Tag des Monats und Uhrzeit definiert ist (beispielsweise `Feb 1 06:25:43`). Für diese Daten können Sie das folgende Muster definieren:

```
SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
```

> **Note**
>
> Grok-Muster können immer nur eine Zeile auf einmal verarbeiten. Mehrzeilige Muster werden nicht unterstützt. Auch Zeilenumbrüche innerhalb eines Musters werden nicht unterstützt.

Benutzerdefinierte Werte für den Grok-Klassifikator

Wenn Sie einen Grok-Klassifikator definieren, geben Sie die folgenden Werte an, um den benutzerdefinierten Klassifikator zu erstellen.

Name

Name des Classifiers.

Klassifizierung

Die Zeichenfolge, die geschrieben wird, um das Format der klassifizierten Daten zu beschreiben (z. B. `special-logs`).

Grok-Muster

Die Menge von Mustern, die auf die Datenspeicher angewendet werden, um festzustellen, ob es eine Übereinstimmung gibt. Diese Muster stammen von AWS Glue integrierte Muster und alle benutzerdefinierten Muster, die Sie definieren.

Im Folgenden sehen Sie ein Beispiel für ein Grok-Muster:

```
%{TIMESTAMP_ISO8601:timestamp} \[%{MESSAGEPREFIX:message_prefix}\]
 %{CRAWLERLOGLEVEL:loglevel} : %{GREEDYDATA:message}
```

Wenn die Daten mit `TIMESTAMP_ISO8601` übereinstimmen, wird eine Schema-Spalte `timestamp` erzeugt. Das Verhalten ist für die anderen benannten Muster im Beispiel ähnlich.

Benutzerdefinierte Muster

Von Ihnen definierte, optionale Muster. Diese Muster werden durch das Grok-Muster referenziert, das Ihre Daten klassifiziert. Sie können diese angepassten Muster in dem Grok-Muster referenzieren, das auf Ihre Daten angewendet wird. Jedes angepasste Komponentenmuster muss sich in einer separaten Zeile befinden. Die Syntax für reguläre Ausdrücke (regex) wird verwendet, um das Muster zu definieren.

Im Folgenden finden Sie ein Beispiel für die Verwendung von angepassten Mustern:

```
CRAWLERLOGLEVEL (BENCHMARK|ERROR|WARN|INFO|TRACE)
MESSAGEPREFIX .*-.*-.*-.*-.*
```

Das erste angepasste benannte Muster (`CRAWLERLOGLEVEL`) ist eine Übereinstimmung, wenn die Daten mit einer der aufgezählten Zeichenfolgen übereinstimmen. Das zweite benutzerdefinierte Muster (`MESSAGEPREFIX`) versucht, eine Nachrichtenpräfix-Zeichenfolge zu finden.

AWS Glue verfolgt die Erstellungszeit, die Uhrzeit der letzten Aktualisierung und die Version Ihres Klassifikators.

Integrierte Muster

AWS Glue bietet viele gängige Muster, mit denen Sie einen benutzerdefinierten Klassifikator erstellen können. Sie fügen dem `grok pattern` in einer Classifier-Definition ein benanntes Muster hinzu.

Die folgende Liste umfasst einer Zeile für jedes Muster. Pro Zeile folgt nach der Definition der Name des Musters. Die Syntax für reguläre Ausdrücke (regex) wird bei der Definition des Musters verwendet.

```
#<noloc>&GLU;</noloc> Built-in patterns
 USERNAME [a-zA-Z0-9._-]+
```

```
 USER %{USERNAME:UNWANTED}
 INT (?:[+-]?(?:[0-9]+))
 BASE10NUM (?<![0-9.+-])(?>[+-]?(?:(?:[0-9]+(?:\.[0-9]+)?)|(?:\.[0-9]+)))
 NUMBER (?:%{BASE10NUM:UNWANTED})
 BASE16NUM (?<![0-9A-Fa-f])(?:[+-]?(?:0x)?(?:[0-9A-Fa-f]+))
 BASE16FLOAT \b(?<![0-9A-Fa-f.])(?:[+-]?(?:0x)?(?:(?:[0-9A-Fa-f]+(?:\.[0-9A-Fa-f]*)?)|
(?:\.[0-9A-Fa-f]+)))\b
 BOOLEAN (?i)(true|false)

 POSINT \b(?:[1-9][0-9]*)\b
 NONNEGINT \b(?:[0-9]+)\b
 WORD \b\w+\b
 NOTSPACE \S+
 SPACE \s*
 DATA .*?
 GREEDYDATA .*
 #QUOTEDSTRING (?:(?<!\\)(?:"(?:\\.|[^\\"])*"|(?:'(?:\\.|[^\\'])*')|(?:`(?:\\.|[^\
\`])*`)))
 QUOTEDSTRING (?>(?<!\\)(?>"(?>\\.|[^\\"]+)+"|""|(?>'(?>\\.|[^\\']+)+')|''|(?>`(?>\\.|
[^\\`]+)+`)|``))
 UUID [A-Fa-f0-9]{8}-(?:[A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}

 # Networking
 MAC (?:%{CISCOMAC:UNWANTED}|%{WINDOWSMAC:UNWANTED}|%{COMMONMAC:UNWANTED})
 CISCOMAC (?:(?:[A-Fa-f0-9]{4}\.){2}[A-Fa-f0-9]{4})
 WINDOWSMAC (?:(?:[A-Fa-f0-9]{2}-){5}[A-Fa-f0-9]{2})
 COMMONMAC (?:(?:[A-Fa-f0-9]{2}:){5}[A-Fa-f0-9]{2})
 IPV6 ((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-
Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:
[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.
(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]
{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|
2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|
((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-
f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|
1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:)))(%.+)?
 IPV4 (?<![0-9])(?:(?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2}))(?![0-9])
 IP (?:%{IPV6:UNWANTED}|%{IPV4:UNWANTED})
```

```
 HOSTNAME \b(?:[0-9A-Za-z][0-9A-Za-z-_]{0,62})(?:\.(?:[0-9A-Za-z][0-9A-Za-z-_]
{0,62}))*(\.?|\b)
 HOST %{HOSTNAME:UNWANTED}
 IPORHOST (?:%{HOSTNAME:UNWANTED}|%{IP:UNWANTED})
 HOSTPORT (?:%{IPORHOST}:%{POSINT:PORT})

 # paths
 PATH (?:%{UNIXPATH}|%{WINPATH})
 UNIXPATH (?>/(?>[\w_%!$@:.,~-]+|\\.)*)+
 #UNIXPATH (?<![\w\/])(?:/[^\/\s?*]*)+
 TTY (?:/dev/(pts|tty([pq])?)(\w+)?/?(?:[0-9]+))
 WINPATH (?>[A-Za-z]+:|\\)(?:\\[^\\?*]*)+
 URIPROTO [A-Za-z]+(\+[A-Za-z+]+)?
 URIHOST %{IPORHOST}(?::%{POSINT:port})?
 # uripath comes loosely from RFC1738, but mostly from what Firefox
 # doesn't turn into %XX
 URIPATH (?:/[A-Za-z0-9$.+!*'(){},~:;=@#%_\-]*)+
 #URIPARAM \?(?:[A-Za-z0-9]+(?:=(?:[^&]*))?(?:&(?:[A-Za-z0-9]+(?:=(?:[^&]*))?)?)*)?
 URIPARAM \?[A-Za-z0-9$.+!*'|(){},~@#%&/=:;_?\-\[\]]*
 URIPATHPARAM %{URIPATH}(?:%{URIPARAM})?
 URI %{URIPROTO}://(?:%{USER}(?::[^@]*)?@)?(?:%{URIHOST})?(?:%{URIPATHPARAM})?

 # Months: January, Feb, 3, 03, 12, December
 MONTH \b(?:Jan(?:uary)?|Feb(?:ruary)?|Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|
Aug(?:ust)?|Sep(?:tember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)\b
 MONTHNUM (?:0?[1-9]|1[0-2])
 MONTHNUM2 (?:0[1-9]|1[0-2])
 MONTHDAY (?:(?:0[1-9])|(?:[12][0-9])|(?:3[01])|[1-9])

 # Days: Monday, Tue, Thu, etc...
 DAY (?:Mon(?:day)?|Tue(?:sday)?|Wed(?:nesday)?|Thu(?:rsday)?|Fri(?:day)?|
Sat(?:urday)?|Sun(?:day)?)

 # Years?
 YEAR (?>\d\d){1,2}
 # Time: HH:MM:SS
 #TIME \d{2}:\d{2}(?::\d{2}(?:\.\d+)?)?
 # TIME %{POSINT<24}:%{POSINT<60}(?::%{POSINT<60}(?:\.%{POSINT})?)?
 HOUR (?:2[0123]|[01]?[0-9])
 MINUTE (?:[0-5][0-9])
 # '60' is a leap second in most time standards and thus is valid.
 SECOND (?:(?:[0-5]?[0-9]|60)(?:[:.,][0-9]+)?)
 TIME (?!<[0-9])%{HOUR}:%{MINUTE}(?::%{SECOND})(?![0-9])
 # datestamp is YYYY/MM/DD-HH:MM:SS.UUUU (or something like it)
```

```
  DATE_US %{MONTHNUM}[/-]%{MONTHDAY}[/-]%{YEAR}
  DATE_EU %{MONTHDAY}[./-]%{MONTHNUM}[./-]%{YEAR}
  DATESTAMP_US %{DATE_US}[- ]%{TIME}
  DATESTAMP_EU %{DATE_EU}[- ]%{TIME}
  ISO8601_TIMEZONE (?:Z|[+-]%{HOUR}(?::?%{MINUTE}))
  ISO8601_SECOND (?:%{SECOND}|60)
  TIMESTAMP_ISO8601 %{YEAR}-%{MONTHNUM}-%{MONTHDAY}[T ]%{HOUR}:?%{MINUTE}(?::?
%{SECOND})?%{ISO8601_TIMEZONE}?
  TZ (?:[PMCE][SD]T|UTC)
  DATESTAMP_RFC822 %{DAY} %{MONTH} %{MONTHDAY} %{YEAR} %{TIME} %{TZ}
  DATESTAMP_RFC2822 %{DAY}, %{MONTHDAY} %{MONTH} %{YEAR} %{TIME} %{ISO8601_TIMEZONE}
  DATESTAMP_OTHER %{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{TZ} %{YEAR}
  DATESTAMP_EVENTLOG %{YEAR}%{MONTHNUM2}%{MONTHDAY}%{HOUR}%{MINUTE}%{SECOND}
  CISCOTIMESTAMP %{MONTH} %{MONTHDAY} %{TIME}

  # Syslog Dates: Month Day HH:MM:SS
  SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
  PROG (?:[\w._/%-]+)
  SYSLOGPROG %{PROG:program}(?:\[%{POSINT:pid}\])?
  SYSLOGHOST %{IPORHOST}
  SYSLOGFACILITY <%{NONNEGINT:facility}.%{NONNEGINT:priority}>
  HTTPDATE %{MONTHDAY}/%{MONTH}/%{YEAR}:%{TIME} %{INT}

  # Shortcuts
  QS %{QUOTEDSTRING:UNWANTED}

  # Log formats
  SYSLOGBASE %{SYSLOGTIMESTAMP:timestamp} (?:%{SYSLOGFACILITY} )?%{SYSLOGHOST:logsource}
  %{SYSLOGPROG}:

  MESSAGESLOG %{SYSLOGBASE} %{DATA}

  COMMONAPACHELOG %{IPORHOST:clientip} %{USER:ident} %{USER:auth}
  \[%{HTTPDATE:timestamp}\] "(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/
%{NUMBER:httpversion})?|%{DATA:rawrequest})" %{NUMBER:response} (?:%{Bytes:bytes=
%{NUMBER}|-})
  COMBINEDAPACHELOG %{COMMONAPACHELOG} %{QS:referrer} %{QS:agent}
  COMMONAPACHELOG_DATATYPED %{IPORHOST:clientip} %{USER:ident;boolean} %{USER:auth}
  \[%{HTTPDATE:timestamp;date;dd/MMM/yyyy:HH:mm:ss Z}\] "(?:%{WORD:verb;string}
  %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion;float})?|%{DATA:rawrequest})"
  %{NUMBER:response;int} (?:%{NUMBER:bytes;long}|-)


  # Log Levels
```

```
 LOGLEVEL ([A|a]lert|ALERT|[T|t]race|TRACE|[D|d]ebug|DEBUG|[N|n]otice|NOTICE|[I|i]nfo|
INFO|[W|w]arn?(?:ing)?|WARN?(?:ING)?|[E|e]rr?(?:or)?|ERR?(?:OR)?|[C|c]rit?(?:ical)?|
CRIT?(?:ICAL)?|[F|f]atal|FATAL|[S|s]evere|SEVERE|EMERG(?:ENCY)?|[Ee]merg(?:ency)?)
```

Angepasste XML-Classifier schreiben

XML definiert die Struktur eines Dokuments mithilfe von Tags in der Datei. Mit einem angepassten XML-Classifier können Sie den Tag-Namen angeben, der für die Definition einer Zeile verwendet wird.

Benutzerdefinierte Klassifikatorwerte für einen XML-Klassifikator

Wenn Sie einen XML-Klassifikator definieren, geben Sie die folgenden Werte an AWS Glue um den Klassifikator zu erstellen. Das Klassifizierungsfeld dieses Classifiers ist auf `xml` festgelegt.

Name

Name des Classifiers.

Row-Tag

Der XML-Tag-Name, der eine Tabellenzeile im XML-Dokument definiert, ohne spitze Klammern (< >). Der Name muss den XML-Regeln für ein Tag entsprechen.

> **Note**
>
> Das Element, das die Zeilendaten enthält, darf kein selbstschließendes, leeres Element sein. Dieses leere Element wird beispielsweise nicht analysiert von AWS Glue:
>
> ```
> <row att1=”xx” att2=”yy” />
> ```
>
> Leere Elemente können wie folgt geschrieben werden:
>
> ```
> <row att1=”xx” att2=”yy”> </row>
> ```

AWS Glue verfolgt die Erstellungszeit, die Uhrzeit der letzten Aktualisierung und die Version Ihres Klassifikators.

Angenommen, Sie haben die folgende XML-Datei. Um einen zu erstellen AWS Glue Eine Tabelle, die nur Spalten für Autor und Titel enthält, erstellen Sie einen Klassifikator in AWS Glue Konsole mit dem Zeilen-Tag als`AnyCompany`. Fügen Sie dann einen Crawler hinzu, der den benutzerdefinierten Classifier verwendet, und führen Sie ihn aus.

```
<?xml version="1.0"?>
<catalog>
   <book id="bk101">
     <AnyCompany>
       <author>Rivera, Martha</author>
       <title>AnyCompany Developer Guide</title>
     </AnyCompany>
   </book>
   <book id="bk102">
     <AnyCompany>
       <author>Stiles, John</author>
       <title>Style Guide for AnyCompany</title>
     </AnyCompany>
   </book>
</catalog>
```

Angepasste JSON-Classifier schreiben

JSON ist ein Datenaustauschformat. Es definiert Datenstrukturen mit Name-Wert-Paaren oder einer geordneten Liste von Werten. Mit einem angepassten JSON-Classifier können Sie den JSON-Pfad zu einer Datenstruktur angeben, über den das Schema für Ihre Tabelle definiert wird.

Benutzerdefinierte Klassifiziererwerte in AWS Glue

Wenn Sie einen JSON-Klassifikator definieren, geben Sie die folgenden Werte an AWS Glue um den Klassifikator zu erstellen. Das Klassifizierungsfeld dieses Classifiers ist auf `json` festgelegt.

Name

Name des Classifiers.

JSON-Pfad

Ein JSON-Pfad, der auf ein Objekt zeigt, das zur Definition eines Tabellenschemas verwendet wird. Der JSON-Pfad kann in Punkt- oder Klammer-Notation angegeben werden. Folgende Operatoren werden unterstützt:

Beschreibung
Root-Element eines JSON-Objekts. Beginn für alle Pfadausdrücke
Platzhalterzeichen. Überall dort möglich, wo ein Name oder eine Zahl im JSON-Pfad benötigt wird.
Untergeordnetes Element in Punkt-Notation. Gibt ein untergeordnetes Feld in einem JSON-Objekt an.
Untergeordnetes Element in Klammer-Notation. Gibt ein untergeordnetes Feld in einem JSON-Objekt an. Es kann nur ein einziges untergeordnetes Feld angegeben werden.
Array-Index. Gibt den Wert eines Arrays über den Index an.

AWS Glue verfolgt die Erstellungszeit, die Uhrzeit der letzten Aktualisierung und die Version Ihres Klassifikators.

Example der Verwendung eines JSON-Classifiers zum Auslesen von Datensätzen aus einem Array

Angenommen, Ihre JSON-Daten sind ein Array mit Datensätzen. Die ersten paar Zeilen Ihrer Datei könnten beispielsweise wie folgt aussehen:

```
[
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ak",
    "name": "Alaska"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:1",
```

```
    "name": "Alabama's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:2",
    "name": "Alabama's 2nd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:3",
    "name": "Alabama's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:4",
    "name": "Alabama's 4th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:5",
    "name": "Alabama's 5th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:6",
    "name": "Alabama's 6th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:7",
    "name": "Alabama's 7th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:1",
    "name": "Arkansas's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:2",
    "name": "Arkansas's 2nd congressional district"
  },
  {
    "type": "constituency",
```

```
    "id": "ocd-division\/country:us\/state:ar\/cd:3",
    "name": "Arkansas's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:4",
    "name": "Arkansas's 4th congressional district"
  }
]
```

Bei der Ausführung eines Crawlers mit dem integrierten JSON-Classifier wird die gesamte Datei zur Definition des Schemas verwendet. Da Sie keinen JSON-Pfad angeben, behandelt der Crawler die Daten als ein Objekt (als Array). Das Schema kann beispielsweise wie folgt aussehen:

```
root
|-- record: array
```

Um ein Schema zu erstellen, das auf jedem Datensatz im JSON-Array basiert, erstellen Sie einen angepassten JSON-Classifier und geben Sie den JSON-Pfad als `$[*]` an. Wenn Sie diesen JSON-Pfad angeben, fragt der Classifier alle 12 Datensätze im Array ab, um das Schema zu bestimmen. Das resultierende Schema enthält für jedes Objekt eigene Felder, wie im folgenden Beispiel gezeigt:

```
root
|-- type: string
|-- id: string
|-- name: string
```

Example Verwendung eines JSON-Classifiers, um nur Teile einer Datei zu untersuchen

Angenommen, Ihre JSON-Daten folgen dem Muster der JSON-Beispieldatei `s3://awsglue-datasets/examples/us-legislators/all/areas.json` von http://everypolitician.org/. Beispielobjekte in der JSON-Datei sehen wie folgt aus:

```
{
  "type": "constituency",
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
```

```
{
  "type": "constituency",
  "identifiers": [
    {
      "scheme": "dmoz",
      "identifier": "Regional\/North_America\/United_States\/Alaska\/"
    },
    {
      "scheme": "freebase",
      "identifier": "\/m\/0hjy"
    },
    {
      "scheme": "fips",
      "identifier": "US02"
    },
    {
      "scheme": "quora",
      "identifier": "Alaska-state"
    },
    {
      "scheme": "britannica",
      "identifier": "place\/Alaska"
    },
    {
      "scheme": "wikidata",
      "identifier": "Q797"
    }
  ],
  "other_names": [
    {
      "lang": "en",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "fr",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "nov",
      "note": "multilingual",
      "name": "Alaska"
    }
```

```
  ],
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
```

Bei der Ausführung eines Crawlers mit dem integrierten JSON-Classifier wird die gesamte Datei zum Erstellen des Schemas verwendet. Es könnte das folgende Schema erstellt werden:

```
root
|-- type: string
|-- id: string
|-- name: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
```

Um ein Schema zu erstellen, das nur das "`id`"-Objekt verwendet, erstellen Sie einen angepassten JSON-Classifier und geben Sie den JSON-Pfad als `$.id` an. Dann basiert das Schema nur auf dem Feld "`id`":

```
root
|-- record: string
```

Die ersten paar mit diesem Schema extrahierten Datenzeilen sehen so aus:

```
{"record": "ocd-division/country:us/state:ak"}
{"record": "ocd-division/country:us/state:al/cd:1"}
{"record": "ocd-division/country:us/state:al/cd:2"}
{"record": "ocd-division/country:us/state:al/cd:3"}
{"record": "ocd-division/country:us/state:al/cd:4"}
{"record": "ocd-division/country:us/state:al/cd:5"}
```

```
{"record": "ocd-division/country:us/state:al/cd:6"}
{"record": "ocd-division/country:us/state:al/cd:7"}
{"record": "ocd-division/country:us/state:ar/cd:1"}
{"record": "ocd-division/country:us/state:ar/cd:2"}
{"record": "ocd-division/country:us/state:ar/cd:3"}
{"record": "ocd-division/country:us/state:ar/cd:4"}
{"record": "ocd-division/country:us/state:as"}
{"record": "ocd-division/country:us/state:az/cd:1"}
{"record": "ocd-division/country:us/state:az/cd:2"}
{"record": "ocd-division/country:us/state:az/cd:3"}
{"record": "ocd-division/country:us/state:az/cd:4"}
{"record": "ocd-division/country:us/state:az/cd:5"}
{"record": "ocd-division/country:us/state:az/cd:6"}
{"record": "ocd-division/country:us/state:az/cd:7"}
```

Um ein Schema zu erstellen, das auf einem tief verschachtelten Objekt (z. B. "`identifier`") in der JSON-Datei basiert, können Sie einen angepassten JSON-Classifier erstellen und den JSON-Pfad als `$.identifiers[*].identifier` angeben. Obwohl das Schema dem vorherigen Beispiel ähnlich ist, basiert es auf einem anderen Objekt in der JSON-Datei.

Das Schema sieht wie folgt aus:

```
root
|-- record: string
```

Die ersten Datenzeilen aus der Tabelle zeigen, dass das Schema auf den Daten im Objekt "`identifier`" basiert:

```
{"record": "Regional/North_America/United_States/Alaska/"}
{"record": "/m/0hjy"}
{"record": "US02"}
{"record": "5879092"}
{"record": "4001016-8"}
{"record": "destination/alaska"}
{"record": "1116270"}
{"record": "139487266"}
{"record": "n79018447"}
{"record": "01490999-8dec-4129-8254-eef6e80fadc3"}
{"record": "Alaska-state"}
{"record": "place/Alaska"}
```

```
{"record": "Q797"}
{"record": "Regional/North_America/United_States/Alabama/"}
{"record": "/m/0gyh"}
{"record": "US01"}
{"record": "4829764"}
{"record": "4084839-5"}
{"record": "161950"}
{"record": "131885589"}
```

Um eine Tabelle zu erstellen, die auf einem anderen tief verschachtelten Objekt basiert (z. B. dem Feld "name" im Array "other_names" in der JSON-Datei), können Sie einen angepassten JSON-Classifier erstellen und den JSON-Pfad als $.other_names[*].name festlegen. Obwohl das Schema dem vorherigen Beispiel ähnlich ist, basiert es auf einem anderen Objekt in der JSON-Datei. Das Schema sieht wie folgt aus:

```
root
|-- record: string
```

Die ersten Datenzeilen aus der Tabelle zeigen, dass das Schema auf den Daten im Objekt "name" im Array "other_names" basiert:

```
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Аляска"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "######"}
{"record": "######"}
{"record": "######"}
{"record": "Alaska"}
{"record": "Alyaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Штат Аляска"}
{"record": "Аляска"}
{"record": "Alaska"}
```

```
{"record": "#######"}
```

Angepasste CSV-Classifier schreiben

Mit benutzerdefinierten CSV-Klassifikatoren können Sie Datentypen für jede Spalte im benutzerdefinierten CSV-Klassifikatorfeld angeben. Sie können den Datentyp jeder Spalte durch ein Komma trennen. Durch die Angabe von Datentypen können Sie die vom Crawler abgeleiteten Datentypen überschreiben und sicherstellen, dass die Daten entsprechend klassifiziert werden.

Sie können den SerDe CSV-Wert für die Verarbeitung im Klassifikator festlegen, der im Datenkatalog angewendet wird.

Wenn Sie einen benutzerdefinierten Klassifikator erstellen, können Sie den Classifer auch für verschiedene Crawler wiederverwenden.

- Bei CSV-Dateien, die nur Kopfzeilen (keine Daten) enthalten, werden diese Dateien als UNKNOWN klassifiziert, da nicht genügend Informationen bereitgestellt werden. Wenn Sie in der Option Column headings (Spaltenüberschriften) angeben, dass die CSV-Datei „Has headings“, also Überschriften enthält, und die Datentypen angeben, können wir diese Dateien korrekt klassifizieren.

Sie können einen benutzerdefinierten CSV-Classifier zum Ableiten des Schemas verschiedener Typen von CSV-Daten verwenden. Zu den benutzerdefinierten Attributen, die Sie für Ihren Klassifikator angeben können, gehören Trennzeichen, eine SerDe CSV-Option, Optionen für den Header und die Angabe, ob bestimmte Validierungen an den Daten durchgeführt werden sollen.

Benutzerdefinierte Klassifiziererwerte in AWS Glue

Wenn Sie einen CSV-Klassifikator definieren, geben Sie die folgenden Werte an AWS Glue um den Klassifikator zu erstellen. Das Klassifizierungsfeld dieses Classifiers ist auf `csv` festgelegt.

Name des Classifiers

Name des Classifiers.

CSV-Serde

Legt den CSV-Wert SerDe für die Verarbeitung im Klassifikator fest, der im Datenkatalog angewendet wird. Die Optionen sind Open CSV SerDe, Lazy SerDe Simple und Keine. Sie können den Wert „None“ angeben, wenn der Crawler die Erkennung durchführen soll.

Spaltentrennzeichen

Eine benutzerdefiniertes Symbol zur Bezeichnung, wodurch die einzelnen Spalteneinträge in der Zeile voneinander getrennt werden. Geben Sie ein Unicode-Zeichen an. Wenn Sie Ihr Trennzeichen nicht eingeben können, können Sie es kopieren und einfügen. Dies funktioniert für druckbare Zeichen, einschließlich solcher, die Ihr System nicht unterstützt (normalerweise angezeigt als □).

Anführungszeichen

Ein benutzerdefiniertes Symbol zur Bezeichnung, wodurch Inhalte zu einem einzelnen Spaltenwert miteinander kombiniert werden. Muss sich von dem Spaltentrennzeichen unterscheiden. Geben Sie ein Unicode-Zeichen an. Wenn Sie Ihr Trennzeichen nicht eingeben können, können Sie es kopieren und einfügen. Dies funktioniert für druckbare Zeichen, einschließlich solcher, die Ihr System nicht unterstützt (normalerweise angezeigt als □).

Spaltenüberschriften

Gibt das Verhalten an, wie Spaltenüberschriften in der CSV-Datei erkannt werden sollen. Wenn Ihre benutzerdefinierte CSV-Datei Spaltenüberschriften besitzt, geben Sie eine durch Komma getrennte Liste der Spaltenüberschriften ein.

Verarbeitungsoptionen: Dateien mit einzelnen Spalte erlauben

Aktiviert die Verarbeitung von Dateien, die nur eine Spalte enthalten.

Verarbeitungsoptionen: Leerzeichen vor Identifizierung von Spaltenwerten abtrennen

Gibt an, ob Werte vor dem Identifizieren des Typ der Spaltenwerte abgetrennt werden sollen.

Benutzerdefinierte Datentypen – optional

Geben Sie den benutzerdefinierten Datentyp durch ein Komma getrennt ein. Gibt die benutzerdefinierten Datentypen in der CSV-Datei an. Der benutzerdefinierte Datentyp muss ein unterstützter Datentyp sein. Unterstützte Datentypen sind: „BINARY“, „BOOLEAN“, „DATE“, „DECIMAL“, „DOUBLE“, „FLOAT“, „INT“, „LONG“, „SHORT“, „STRING“, „TIMESTAMP“. Bei nicht unterstützten Datentypen wird ein Fehler angezeigt.

Erstellen von Klassifikatoren mit dem AWS Glue Konsole

Ein Classifier bestimmt das Schema Ihrer Daten. Sie können einen benutzerdefinierten Classifier erstellen und von AWS Glue aus auf ihn verweisen.

Classifier erstellen

Um einen Klassifikator in der AWS Glue Konsole, wählen Sie Klassifikator hinzufügen. Wenn Sie einen Classifier definieren, geben Sie Werte für Folgendes an:

- Classifier-Name – Geben Sie einen eindeutigen Namen für Ihren Classifier ein.
- Classifier-Typ – Der Klassifizierungstyp der Tabellen, die von diesem Classifier abgeleitet werden.
- Letzte Aktualisierung – Der letzte Zeitpunkt, zu dem dieser Classifier aktualisiert wurde.

Name des Classifiers

Geben Sie einen eindeutigen Namen für Ihren Classifier ein.

Classifier-Typ

Wählen Sie den zu erstellenden Classifier aus.

Konfigurieren Sie je nach gewähltem Classifier-Typ die folgenden Eigenschaften für Ihren Classifier:

Grok

- Klassifizierung

  Beschreiben Sie das Format oder den Typ der Daten, die klassifiziert werden, oder geben Sie ein benutzerdefiniertes Label an.

- Grok-Muster

  Dies dient dazu, Ihre Daten zu analysieren und in ein strukturiertes Schema einzufügen. Das Grok-Muster besteht aus benannten Mustern, die das Format Ihres Datenspeichers beschreiben. Sie schreiben dieses Grok-Muster mit den benannten integrierten Mustern, die bereitgestellt werden von AWS Glue und benutzerdefinierte Muster, die Sie schreiben und in das Feld Benutzerdefinierte Muster aufnehmen. Obwohl die Ergebnisse des Grok-Debuggers möglicherweise nicht mit den Ergebnissen von übereinstimmen AWS Glue Genau, wir empfehlen Ihnen, Ihr Muster anhand einiger Beispieldaten mit einem Grok-Debugger auszuprobieren. Grok-Debugger finden Sie im Internet. Die benannten integrierten Muster werden bereitgestellt von AWS Glue sind im Allgemeinen mit Grok-Mustern kompatibel, die im Internet verfügbar sind.

Erstellen Sie Ihr Grok-Muster durch iteratives Hinzufügen von benannten Mustern und überprüfen Sie Ihre Ergebnisse in einem Debugger. Diese Aktivität gibt Ihnen die Gewissheit, dass wenn AWS Glue Der Crawler führt Ihr Grok-Muster aus, Ihre Daten können analysiert werden.

- Benutzerdefinierte Muster

  Für Grok-Classifier sind dies optionale Bausteine für das Grok pattern (Grok-Muster), das Sie schreiben. Wenn integrierte Muster Ihre Daten nicht analysieren können, müssen Sie möglicherweise ein benutzerdefiniertes Muster schreiben. Diese benutzerdefinierten Muster werden in diesem Feld definiert und im Feld Grok pattern (Grok-Muster) referenziert. Jedes benutzerdefinierte Muster wird in einer separaten Zeile definiert. Es besteht wie integrierte Muster aus einer benannten Musterdefinition, die reguläre Ausdruckssyntax (Regex) verwendet.

  Im folgenden Beispiel folgt auf den Namen `MESSAGEPREFIX` eine reguläre Ausdrucksdefinition für Ihre Daten, um festzustellen, ob das Muster eingehalten wird.

  ```
  MESSAGEPREFIX .*-.*-.*-.*-.*
  ```

XML

- Row-Tag

  Für XML-Classifier ist dies der Name des XML-Tags zur Definition einer Tabellenzeile im XML-Dokument. Geben Sie den Namen ohne spitze Klammern < > an. Der Name muss den XML-Regeln für ein Tag entsprechen.

  Weitere Informationen finden Sie unter Angepasste XML-Classifier schreiben.

JSON

- JSON-Pfad

  Für JSON-Classifier handelt es sich hierbei um den JSON-Pfad zum Objekt, Array oder Wert, das bzw. der eine Zeile der Tabelle, die erstellt wird, definiert. Geben Sie den Namen entweder in einer JSON-Syntax mit Punkt oder Klammer ein AWS Glue unterstützte Operatoren.

Weitere Informationen finden Sie in der Liste der Operatoren in Angepasste JSON-Classifier schreiben.

CSV

- Spaltentrennzeichen

  Ein Symbol zur Bezeichnung, wodurch die einzelnen Spalteneinträge in der Zeile voneinander getrennt werden. Wählen Sie das Begrenzungszeichen in der Liste aus, oder wählen Sie `Other`, um ein benutzerdefiniertes Trennzeichen einzugeben.
- Anführungszeichen

  Ein einzelnes Zeichen oder Symbol zur Bezeichnung wodurch Inhalte zu einem einzelnen Spaltenwert miteinander kombiniert werden. Muss sich von dem Spaltentrennzeichen unterscheiden. Wählen Sie das Angebotssymbol aus der Liste aus, oder wählen Sie `Other`, um ein benutzerdefiniertes Anführungszeichen einzugeben.
- Spaltenüberschriften

  Gibt das Verhalten an, wie Spaltenüberschriften in der CSV-Datei erkannt werden sollen. Sie können `Has headings`, `No headings` oder `Detect headings` wählen. Wenn Ihre benutzerdefinierte CSV-Datei Spaltenüberschriften besitzt, geben Sie eine durch Komma getrennte Liste der Spaltenüberschriften ein.
- Zulassen von Dateien mit einzelner Spalte

  Damit die Tabelle als CSV klassifiziert wird, muss das Tabellenschema mindestens zwei Spalten und zwei Datenzeilen aufweisen. Verwenden Sie diese Option, um die Verarbeitung von Dateien zu erlauben, die nur eine Spalte enthalten.
- Leerzeichen vor Identifizierung von Spaltenwerten abtrennen

  Diese Option gibt an, ob Werte vor dem Identifizieren des Typs der Spaltenwerte abgetrennt werden sollen.
- Benutzerdefinierter Datentyp

  (Optional) – Geben Sie benutzerdefinierte Datentypen in eine kommagetrennte Liste ein. Die unterstützten Datentypen sind: „BINARY“, „BOOLEAN“, „DATE“, „DECIMAL“, „DOUBLE“, „FLOAT“, „INT“, „LONG“, „SHORT“, „STRING“, „TIMESTAMP“.
- CSV-Serde

(Optional) — A SerDe für die Verarbeitung von CSV im Classifier, das im Datenkatalog angewendet wird. Wählen Sie `Open CSV SerDe`, `Lazy Simple SerDe`, oder `None` aus. Sie können den `None`-Wert angeben, wenn der Crawler die Erkennung durchführen soll.

Weitere Informationen finden Sie unter Schreiben von benutzerdefinierten Klassifikatoren für verschiedene Datenformate.

Classifier anzeigen

Um eine Liste aller von Ihnen erstellten Klassifikatoren zu sehen, öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/, und wählen Sie den Tab Classifiers aus.

Die Liste zeigt die folgenden Eigenschaften für jeden Classifier an:

- Classifier – Der Name des Classifiers. Beim Erstellen eines Classifiers müssen Sie einen Namen angeben.
- Klassifizierung – Der Klassifizierungstyp der Tabellen, die von diesem Classifier abgeleitet werden.
- Letzte Aktualisierung – Der letzte Zeitpunkt, zu dem dieser Classifier aktualisiert wurde.

Classifier verwalten

Aus der Klassifikatoren-Liste im AWS Glue In der Konsole können Sie Klassifikatoren hinzufügen, bearbeiten und löschen. Um weitere Details über einen Classifier zu sehen, wählen Sie den Classifier-Namen in der Liste aus. Zu den Details gehören die Informationen, die Sie beim Erstellen des Classifiers definiert haben.

Konfiguration eines Crawlers

Ein Crawler greift auf Ihren Datenspeicher zu, identifiziert Metadaten und erstellt Tabellendefinitionen im. AWS Glue Data Catalog Im Bereich Crawler in der AWS Glue Konsole sind alle Crawler aufgeführt, die Sie erstellen. In der Liste werden Status und Metriken aus der letzten Ausführung Ihres Crawlers angezeigt.

Dieses Thema behandelt den step-by-step Prozess der Konfiguration eines Crawlers und behandelt wichtige Aspekte wie das Einrichten der Crawler-Parameter, das Definieren der zu durchforstenden Datenquellen, das Einrichten der Sicherheit und das Verwalten der gecrawlten Daten.

Themen

- Schritt 1: Festlegen der Crawler-Eigenschaften
- Schritt 2: Auswahl von Datenquellen und Classifier
- Schritt 3: Konfigurieren der Sicherheitseinstellungen
- Schritt 4: Festlegen von Ausgabe und Zeitplanung
- Schritt 5: Überprüfen und Erstellen

Schritt 1: Festlegen der Crawler-Eigenschaften

Um einen Crawler zu konfigurieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/. Wählen Sie im Navigationsbereich Crawlers aus.
2. Wählen Sie Crawler erstellen aus und befolgen Sie die Anweisungen im Assistenten zum Hinzufügen eines Crawlers. Der Assistent führt Sie durch die Schritte, die zum Erstellen eines Crawlers erforderlich sind. Wenn Sie benutzerdefinierte Klassifikatoren hinzufügen möchten, um das Schema zu definieren, finden Sie weitere Informationen unter. Klassifikatoren definieren und verwalten
3. Geben Sie einen Namen für Ihren Crawler und eine Beschreibung ein (optional). Sie können Ihren Crawler mit einem Tag-Schlüssel und einem optionalen Tag-Wert markieren. Nach der Erstellung sind Tag-Schlüssel schreibgeschützt. Verwenden Sie Tags für manche Ressourcen, damit sie leichter zu organisieren und identifizieren sind. Weitere Informationen finden Sie unter AWS Tags unter. AWS Glue

   Name

   Der Name darf Buchstaben (A–Z), Zahlen (0–9), Bindestriche (-) oder Unterstriche (_) enthalten und bis zu 255 Zeichen lang sein.

   Beschreibung

   Die Beschreibung kann bis zu 2048 Zeichen lang sein.

   Tags

   Verwenden Sie Tags zum Organisieren und Identifizieren Ihrer Ressourcen. Weitere Informationen finden Sie hier:

   - AWS Schlagworte in AWS Glue

Schritt 2: Auswahl von Datenquellen und Classifier

Als Nächstes konfigurieren Sie die Datenquellen und Klassifikatoren für den Crawler.

Weitere Informationen zu unterstützten Datenquellen finden Sie unter. Unterstützte Datenquellen für das Crawlen

Konfiguration der Datenquelle

Wählen Sie die entsprechende Option für Sind Ihre Daten bereits zugeordnet AWS Glue Tabellen? wählen Sie „Noch nicht“ oder „Ja“. Standardmäßig ist „Noch nicht“ ausgewählt.

Der Crawler kann als Quelle des Crawls direkt auf Datenspeicher zugreifen oder vorhandene Tabellen im Data Catalog als Quelle verwenden. Wenn der Crawler vorhandene Katalogtabellen verwendet, durchsucht er die Datenspeicher, die durch diese Katalogtabellen angegeben werden.

- Noch nicht: Wählen Sie eine oder mehrere Datenquellen aus, die gecrawlt werden sollen. Ein Crawler kann mehrere Datenspeicher verschiedener Arten (Amazon S3, JDBC etc.) durchsuchen.

  Sie können jeweils nur einen Datenspeicher konfigurieren. Nachdem Sie die Verbindungsinformationen angegeben und Pfade und Ausschlussmuster eingeschlossen haben, haben Sie die Möglichkeit, einen weiteren Datenspeicher hinzuzufügen.

- Ja: Wählen Sie bestehende Tabellen aus AWS Glue Datenkatalog. Die Katalogtabellen geben die Datenspeicher an, die durchsucht werden sollen. Der Crawler kann in einem einzigen Lauf nur Katalogtabellen durchsuchen. Er kann keine anderen Quelltypen untermischen.

  Ein häufiger Grund zur Angabe einer Katalogtabelle als Quelle besteht darin, dass Sie die Tabelle manuell erstellt haben (da Sie die Struktur des Datenspeichers bereits kannten) und Sie möchten, dass ein Crawler die Tabelle auf dem aktuellen Stand hält, wozu das Hinzufügen neuer Partitionen gehört. Weitere Informationen zu anderen Gründen finden Sie unter Aktualisieren von manuell erstellten Data-Catalog-Tabellen mit Crawlern.

  Wenn Sie als Crawler-Quelltyp vorhandene Tabellen angeben, gelten die folgenden Bedingungen:

  - Der Datenbankname ist optional.
  - Nur Katalogtabellen, die Amazon S3-, Amazon DynamoDB- oder Delta Lake-Datenspeicher angeben, sind zulässig.

- Es werden keine neuen Katalogtabellen erstellt, wenn der Crawler ausgeführt wird. Vorhandene Tabellen werden bei Bedarf aktualisiert. Dazu gehört auch das Hinzufügen neuer Partitionen.
- In den Datenspeichern gefundene gelöschte Objekte werden ignoriert; es werden keine Katalogtabellen gelöscht. Stattdessen schreibt der Crawler eine Protokollmeldung. (`SchemaChangePolicy.DeleteBehavior=LOG`)
- Die Crawler-Konfigurationsoption zum Erstellen eines einzigen Schemas für jeden Amazon-S3-Pfad ist standardmäßig aktiviert und kann nicht deaktiviert werden. (`TableGroupingPolicy=CombineCompatibleSchemas`) Weitere Informationen finden Sie unter Erstellen eines einzelnen Schemas für jeden Amazon S3 S3-Include-Pfad.
- Sie können Katalogtabellen als Quelle nicht mit anderen Arten von Quellen (z. B. Amazon S3 oder Amazon DynamoDB) mischen.

Um Delta-Tabellen zu verwenden, erstellen Sie zunächst eine Delta-Tabelle mit Athena DDL oder der AWS Glue API.

Stellen Sie mit Athena den Speicherort auf Ihren Amazon S3 S3-Ordner und den Tabellentyp auf „DELTA“ ein.

```
CREATE EXTERNAL TABLE database_name.table_name
LOCATION 's3://bucket/folder/'
TBLPROPERTIES ('table_type' = 'DELTA')
```

Geben Sie mithilfe der AWS Glue API den Tabellentyp in der Tabelle mit den Tabellenparametern an. Die Tabellenparameter müssen das folgende Schlüssel/Wert-Paar enthalten. Weitere Informationen zum Erstellen einer Tabelle finden Sie in der Boto3-Dokumentation für create_table.

```
{
    "table_type":"delta"
}
```

Datenquellen

Wählen Sie die Liste der Datenquellen aus, die vom Crawler gescannt werden sollen, oder fügen Sie diese hinzu.

(Optional) Wenn Sie JDBC als Datenquelle wählen, können Sie Ihre eigenen JDBC-Treiber verwenden, wenn Sie den Verbindungszugriff angeben, in dem die Treiberinformationen gespeichert werden.

Pfad einschließen

Wenn es darum geht, zu beurteilen, was bei einem Crawl ein- bzw. ausgeschlossen werden soll, beginnt der Crawler mit der Bewertung des erforderlichen Include-Pfads. Für Amazon S3, MongoDB, MongoDB Atlas, Amazon DocumentDB (mit MongoDB-Kompatibilität) und relationale Datenspeicher müssen Sie einen Include-Pfad angeben.

Für einen Amazon-S3-Datenspeicher

Wählen Sie aus, ob Sie einen Pfad in diesem Konto oder in einem anderen Konto angeben möchten, und suchen Sie dann nach einem Amazon-S3-Pfad.

Für Amazon-S3-Datenspeicher ist die Include-Pfad-Syntax `bucket-name/folder-name/file-name.ext`. Um alle Objekte in einem Bucket zu durchsuchen, geben Sie nur den Bucket-Namen im Include-Pfad an. Ein Exclude-Muster bezieht sich auf den Include-Pfad.

Für einen Delta Lake-Datenspeicher

Geben Sie einen oder mehrere Amazon S3 S3-Pfade zu Delta-Tabellen als s3://*bucket*/*prefix*/an*object*.

Für einen Iceberg- oder Hudi-Datenspeicher

Geben Sie einen oder mehrere Amazon S3 S3-Pfade, die Ordner mit Iceberg- oder Hudi-Tabellenmetadaten enthalten, als s3:///*bucket*an. *prefix*

Bei Iceberg- und Hudi-Datenspeichern befindet sich der Ordner Iceberg/Hudi möglicherweise in einem untergeordneten Ordner des Stammordners. Der Crawler durchsucht alle Ordner unterhalb eines Pfades nach einem Hudi-Ordner.

Für einen JDBC-Datenspeicher

Geben Sie *<database>* je nach *<schema>* Datenbankprodukt/*<table>*oder*<database>*/*<table>*ein. Oracle Database und MySQL unterstützen kein Schema im Pfad. Sie können das Prozentzeichen (%) durch *<schema>* oder ersetzen*<table>*. Geben Sie beispielsweise für eine Oracle-Datenbank mit einer Systemkennung (SID) von `orcl` `orcl/%` ein, um alle Tabellen zu importieren, auf die der in der Verbindung benannte Benutzer Zugriff hat.

> **Important**
>
> Bitte beachten Sie die Groß- und Kleinschreibung.

> **Note**
>
> Wenn Sie Ihre eigenen JDBC-Treiberversionen verwenden möchten, AWS Glue Crawler verbrauchen Ressourcen in AWS Glue Jobs und Amazon S3 S3-Buckets, um sicherzustellen, dass Ihr bereitgestellter Treiber in Ihrer Umgebung ausgeführt wird. Der zusätzliche Ressourcenverbrauch wird in Ihrem Konto angezeigt. Treiber sind auf die unter Hinzufügen eines beschriebenen Eigenschaften beschränkt AWS Glue Verbindung.

Für einen MongoDB-, MongoDB-Atlas- oder Amazon-DocumentDB-Datenspeicher

Für MongoDB, MongoDB Atlas und Amazon DocumentDB (mit MongoDB-Kompatibilität) ist die Syntax `database/collection`.

Bei JDBC-Datenspeichern lautet die Syntax entweder `database-name/schema-name/table-name` oder `database-name/table-name`. Die Syntax hängt davon ab, ob die Datenbank-Engine Schemas innerhalb einer Datenbank unterstützt. Bei Datenbank-Engines wie MySQL oder Oracle geben Sie beispielsweise keinen `schema-name` im Include-Pfad an. Sie können das Prozentzeichen (%) für ein Schema oder eine Tabelle im Include-Pfad ersetzen, um alle Schemas oder alle Tabellen einer Datenbank darzustellen. Sie können das Prozentzeichen (%) für die Datenbank im Include-Pfad nicht ersetzen.

Maximale Transversaltiefe (nur für Iceberg- oder Hudi-Datenspeicher)

Definiert die maximale Tiefe des Amazon-S3-Pfads, die der Crawler durchqueren kann, um den Iceberg- oder Hudi-Metadatenordner in Ihrem Amazon-S3-Pfad zu erkennen. Der Zweck dieses Parameters besteht darin, die Crawler-Laufzeit zu begrenzen. Der Standardwert beträgt 10 und der Höchstwert beträgt 20.

Exclude-Muster

Diese ermöglichen es Ihnen, bestimmte Dateien oder Tabellen aus dem Crawl auszuschließen. Der Exclude-Pfad bezieht sich auf den Include-Pfad. Um beispielsweise eine Tabelle in Ihrem JDBC-Datenspeicher auszuschließen, geben Sie den Tabellennamen in den Exclude-Pfad ein.

Ein Crawler stellt über einen eine Verbindung zu einem JDBC-Datenspeicher her AWS Glue Verbindung, die eine JDBC-URI-Verbindungszeichenfolge enthält. Der Crawler hat nur Zugriff auf Objekte in der Datenbank-Engine, indem er den JDBC-Benutzernamen und das JDBC-Kennwort in der AWS Glue Verbindung. Der Crawler kann nur Tabellen erstellen, auf die er über die JDBC-Verbindung zugreifen kann. Nachdem der Crawler auf die Datenbank-Engine mit dem JDBC-URI zugegriffen hat, wird der Include-Pfad verwendet, um festzulegen, welche Tabellen der Datenbank-Engine im Data Catalog erstellt werden. Wenn Sie beispielsweise bei MySQL einen Include-Pfad von `MyDatabase/%` angeben, werden alle Tabellen in `MyDatabase` im Data Catalog angelegt. Wenn Sie beim Zugriff auf Amazon Redshift einen Include-Pfad von `MyDatabase/%` angeben, werden alle Tabellen innerhalb aller Schemas für die Datenbank `MyDatabase` im Data Catalog angelegt. Wenn Sie einen Include-Pfad von `MyDatabase/MySchema/%` angeben, werden alle Tabellen in der Datenbank `MyDatabase` sowie im Schema `MySchema` erstellt.

Nachdem Sie einen Include-Pfad angegeben haben, können Sie Objekte aus dem Crawl-Vorgang ausschließen, die Ihr Include-Pfad andernfalls einschließen würde, indem Sie ein oder mehrere `glob`-Exclude-Muster im Unix-Stil angeben. Diese Muster werden auf Ihren Include-Pfad angewandt, um zu ermitteln, welche Objekte ausgeschlossen werden. Diese Muster werden auch als Eigenschaft von Tabellen gespeichert, die vom Crawler erstellt wurden. AWS Glue PySpark Erweiterungen`create_dynamic_frame.from_catalog`, wie z. B. das Lesen der Tabelleneigenschaften und das Ausschließen von Objekten, die durch das Ausschlussmuster definiert sind.

AWS Glue unterstützt die folgenden `glob` Muster im Ausschlussmuster.

Exclude-Muster	Beschreibung
`*.csv`	Entspricht einem Amazon-S3-Pfad, der einen Objektnamen im aktuellen Ordner darstellt, der mit `.csv` endet
`*.*`	Entspricht allen Objektnamen, die einen Punkt enthalten
`*.{csv,avro}`	Entspricht Objektnamen, die mit `.csv` oder `.avro` enden

Exclude-Muster	Beschreibung
`foo.?`	Entspricht Objektnamen, die mit `foo.` beginnen und von einer Erweiterung aus einem Zeichen gefolgt werden
`myfolder/*`	Entspricht den Objekten in einer Ebene des Unterordners von `myfolder`, z. B. `/myfolder/mysource`
`myfolder/*/*`	Entspricht den Objekten in zwei Ebenen von Unterordnern von `myfolder`, z. B. `/myfolder/mysource/data`
`myfolder/**`	Entspricht den Objekten in allen Unterordnern von `myfolder`, z. B. `/myfolder/mysource/mydata` und `/myfolder/mysource/data`
`myfolder**`	Entspricht dem Unterordner `myfolder` und Dateien unter `myfolder`, wie z. B. `/myfolder` und `/myfolder/mydata.txt`
`Market*`	Entspricht Tabellen in einer JDBC-Datenbank mit Namen, die mit `Market` beginnen, z. B. `Market_us` und `Market_fr`

AWS Glue interpretiert `glob` Ausschlussmuster wie folgt:

- Der Schrägstrich (`/`) ist das Begrenzungszeichen zum Trennen von Amazon-S3-Schlüsseln in einer Ordnerhierarchie.
- Das Sternchen (`*`) entspricht einem oder mehreren Zeichen einer Namenskomponente ohne Überschreiten der Ordnergrenzen.
- Ein doppeltes Sternchen (`**`) entspricht keinem oder mehreren Zeichen bei Überschreiten von Ordner- oder Schemagrenzen.
- Das Fragezeichen (`?`) entspricht genau einem Zeichen einer Namenskomponente.

- Der Backslash (\) wird verwendet, um Zeichen zu maskieren, die andernfalls als Sonderzeichen interpretiert werden könnten. Der Ausdruck `\\` entspricht einem einzelnen Backslash und `\{` entspricht einer linken geschweiften Klammer.
- Eckige Klammern `[ ]` erstellen einen Ausdruck in eckigen Klammern, der für ein einzelnes Zeichen einer Namenskomponente aus einer Reihe von Zeichen steht. Beispiel: `[abc]` entspricht `a`, `b` oder `c`. Der Bindestrich (`-`) kann verwendet werden, um einen Bereich anzugeben. So gibt `[a-z]` einen Bereich von `a` bis `z` (einschließlich) an. Diese Formen können auch kombiniert werden, sodass [`abce-g`] für `ab`, `c`, `e`, `f` oder `g` steht. Wenn das Zeichen nach der eckigen Klammer (`[`) ein Ausrufezeichen (`!`) ist, wird der Ausdruck negiert. Zum Beispiel entspricht `[!a-c]` jedem Zeichen außer `a`, `b` oder `c`.

  Innerhalb eines Ausdrucks in eckigen Klammern stehen die Zeichen `*`, `?` und `\` für sich selbst. Der Bindestrich (`-`) steht für sich selbst, wenn er das erste Zeichen innerhalb der Klammern ist oder wenn er das erste Zeichen nach dem `!` ist, wenn Sie negieren.
- Geschweifte Klammern (`{ }`) umschließen eine Gruppe von Untermustern, wobei die Gruppe übereinstimmt, wenn ein Untermuster in der Gruppe übereinstimmt. Ein Komma (`,`) wird verwendet, um Untermuster voneinander zu trennen. Gruppen können nicht verschachtelt werden.
- Führende Punktzeichen in Dateinamen werden als normale Zeichen bei Abgleichsoperationen behandelt. Beispielsweise stimmt das `*`-Exclude-Muster mit dem Dateinamen `.hidden` überein.

Example Exclude-Muster in Amazon S3

Jedes Exclude-Muster wird mit dem Include-Pfad abgeglichen. Angenommen, Sie haben die folgende Amazon-S3-Verzeichnisstruktur:

```
/mybucket/myfolder/
   departments/
      finance.json
      market-us.json
      market-emea.json
      market-ap.json
   employees/
      hr.json
      john.csv
      jane.csv
      juan.txt
```

Beim Include-Pfad `s3://mybucket/myfolder/` sind folgende Ergebnisse Beispiele für Exclude-Muster:

Exclude-Muster	Ergebnisse
`departments/**`	Schließt alle Dateien und Ordner unterhalb von `departments` aus und schließt den Ordner `employees` und dessen Dateien ein.
`departments/market*`	Schließt `market-us.json` , `market-emea.json` und `market-ap.json` aus.
`**.csv`	Schließt alle Objekte unterhalb von `myfolder` aus, deren Namen mit `.csv` endet.
`employees/*.csv`	Schließt alle `.csv`-Dateien im Ordner`employees` aus.

Example Ausschließen einer Teilmenge von Amazon-S3-Partitionen

Angenommen, Ihre Daten werden nach Tagen partitioniert, sodass sich jeder Tag eines Jahres in einer separaten Amazon-S3-Partition befindet. Im Januar 2015 gibt es 31 Partitionen. Um nun die Daten nur für die erste Januarwoche zu durchsuchen, müssen Sie alle Partitionen mit Ausnahme von Tag 1 bis 7 ausschließen:

```
2015/01/{[!0],0[8-9]}**, 2015/0[2-9]/**, 2015/1[0-2]/**
```

Schauen Sie sich die einzelnen Bestandteile dieses "glob"-Musters an. Der erste Teil, `2015/01/{[!0],0[8-9]}**`, schließt alle Tage aus, die nicht mit einer "0" beginnen, sowie Tag 08 und Tag 09 von Monat 01 des Jahres 2015. Beachten Sie, dass "**" als Suffix des Tagesnummernmusters verwendet wird und Ordnergrenzen bis zu Ordnern auf unterer Ebene überschreitet. Bei Verwendung von "*" werden untere Ordnerebenen nicht ausgeschlossen.

Der zweite Teil, `2015/0[2-9]/**`, schließt die Tage in den Monaten 02 bis 09 im Jahr 2015 aus.

Der dritte Teil, `2015/1[0-2]/**`, schließt die Tage in den Monaten 10, 11 und 12 im Jahr 2015 aus.

Example JDBC-Exclude-Muster

Angenommen, Sie führen ein Crawling für eine JDBC-Datenbank mit der folgenden Schemastruktur durch:

```
MyDatabase/MySchema/
   HR_us
   HR_fr
   Employees_Table
   Finance
   Market_US_Table
   Market_EMEA_Table
   Market_AP_Table
```

Beim Include-Pfad `MyDatabase/MySchema/%` sind folgende Ergebnisse Beispiele für Exclude-Muster:

Exclude-Muster	Ergebnisse
`HR*`	Schließt Tabellen mit Namen aus, die mit `HR` beginnen.
`Market_*`	Schließt Tabellen mit Namen aus, die mit `Market_` beginnen.
`**_Table`	Schließt alle Tabellen mit Namen aus, die mit `_Table` enden.

Zusätzliche Crawler-Quellparameter

Jeder Quelltyp benötigt einen anderen Satz zusätzlicher Parameter.

Verbindung

Wählen Sie eine aus oder fügen Sie eine hinzu AWS Glue Verbindung. Weitere Informationen zu Verbindungen finden Sie unter Herstellen einer Verbindung zu Daten.

Zusätzliche Metadaten – optional (für JDBC-Datenspeicher)

Wählen Sie zusätzliche Metadateneigenschaften aus, die der Crawler crawlen soll.

- Kommentare: Crawlen Sie zugehörige Kommentare auf Tabellen- und Spaltenebene.
- Rohdatentypen: Behalten Sie die Rohdatentypen der Tabellenspalten in zusätzlichen Metadaten bei. Als Standardverhalten übersetzt der Crawler die Rohdatentypen in Hive-kompatible Typen.

Klassenname des JDBC-Treibers – optional (für JDBC-Datenspeicher)

Geben Sie einen benutzerdefinierten JDBC-Treiberklassennamen für den Crawler ein, um eine Verbindung zur Datenquelle herzustellen:

- Postgres: org.postgresql.Driver
- MySQL: com.mysql.jdbc.Driver, com.mysql.cj.jdbc.Driver
- Redshift: com.amazon.redshift.jdbc.Driver, com.amazon.redshift.jdbc42.Driver
- Oracle: oracle.jdbc.driver. OracleDriver
- SQL-Server: com.microsoft.sqlserver.jdbc. SQLServerTreiber

S3-Pfad des JDBC-Treibers – optional (für JDBC-Datenspeicher)

Wählen Sie einen vorhandenen Amazon-S3-Pfad zu einer `.jar`-Datei. Hier wird die `.jar`-Datei gespeichert, wenn ein benutzerdefinierter JDBC-Treiber verwendet wird, damit der Crawler eine Verbindung zur Datenquelle herstellt.

Aktivieren Sie Datenabfrage (nur für Amazon-DynamoDB-, MongoDB-, MongoDB-Atlas- und Amazon-DocumentDB-Datenspeicher)

Wählen Sie aus, ob nur eine beispielhafte Datenabfrage durchsucht werden soll. Wenn diese Option nicht ausgewählt ist, wird die gesamte Tabelle durchsucht. Das Scannen aller Datensätze kann eine lange Zeit in Anspruch nehmen, wenn die Tabelle keinen hohen Durchsatz hat.

Erstellen von Tabellen für Abfragen (nur für Delta-Lake-Datenspeicher)

Wählen Sie aus, wie Sie die Delta-Lake-Tabellen erstellen möchten:

- Erstellen nativer Tabellen: Ermöglicht die Integration mit Abfragemodulen, die die direkte Abfrage des Delta-Transaktionsprotokolls unterstützen.
- Erstellen von Symlink-Tabellen: Erstellen Sie basierend auf den angegebenen Konfigurationsparametern einen Symlink-Manifest-Ordner mit Manifest-Dateien, die durch die Partitionsschlüssel partitioniert sind.

Scanrate – optional (nur für DynamoDB-Datenspeicher)

Geben Sie den Prozentsatz der Lesekapazitätseinheiten der DynamoDB-Tabelle an, die vom Crawler verwendet werden sollen. Lesekapazitätseinheiten sind ein von DynamoDB definierter Begriff und ein numerischer Wert, der als Ratenbegrenzer für die Anzahl der Lesevorgänge fungiert, die pro Sekunde für diese Tabelle durchgeführt werden können. Geben Sie einen Wert zwischen 0,1 und 1,5 ein. Falls nicht angegeben, ist der Standardwert 0,5 für bereitgestellte Tabellen und 1/4 der konfigurierten Maximalkapazität für On-Demand-Tabellen. Beachten Sie, dass für Crawler nur der Modus „Bereitgestellte Kapazität“ verwendet werden sollte. AWS Glue

> **Note**
>
> Legen Sie für DynamoDB-Datenspeicher den bereitgestellten Kapazitätsmodus für die Verarbeitung von Lese- und Schreibvorgängen für Ihre Tabellen fest. Der AWS Glue Crawler sollte nicht im On-Demand-Kapazitätsmodus verwendet werden.

Netzwerkverbindung — optional (für Amazon S3-, Delta-, Iceberg-, Hudi- und Catalog-Zieldatenspeicher)

Fügen Sie optional eine Netzwerkverbindung hinzu, die mit diesem Amazon-S3-Ziel verwendet werden soll. Beachten Sie, dass jeder Crawler auf eine Netzwerkverbindung beschränkt ist, sodass alle anderen Amazon-S3-Ziele ebenfalls dieselbe Verbindung verwenden (oder keine, falls leer gelassen).

Weitere Informationen zu Verbindungen finden Sie unter Herstellen einer Verbindung zu Daten.

Nur für eine Untergruppe von Dateien und Stichprobengröße (nur für Amazon-S3-Datenspeicher)

Geben Sie die Anzahl der Dateien in jedem Ordner an, die beim Crawling von Beispieldateien in einem Datensatz durchsucht werden sollen. Wenn dieses Feature aktiviert ist, wählt der Crawler statt alle Dateien in diesem Datensatz nach dem Zufallsprinzip einige Dateien in jedem Ordner aus, bei denen ein Crawling durchgeführt werden muss.

Der Sampling-Crawler eignet sich am besten für Kunden, die bereits über Kenntnisse ihrer Datenformate verfügen und wissen, dass sich Schemas in ihren Ordnern nicht ändern. Wenn Sie dieses Feature aktivieren, wird die Crawler-Laufzeit erheblich reduziert.

Ein gültiger Wert ist eine ganze Zahl zwischen 1 und 249. Wenn nicht angegeben, werden alle Dateien durchsucht.

Nachfolgende Crawler-Ausführungen

Dieses Feld ist ein globales Feld, das sich auf alle Amazon-S3-Datenquellen auswirkt.

- Alle Unterordner crawlen: Crawlen Sie alle Ordner bei jedem weiteren Crawl erneut.
- Nur neue Unterordner crawlen: Nur Amazon-S3-Ordner, die seit dem letzten Crawling hinzugefügt wurden, werden gecrawlt. Wenn die Schemas kompatibel sind, werden den vorhandenen Tabellen neue Partitionen hinzugefügt. Weitere Informationen finden Sie unter the section called “Planung inkrementeller Crawls”.
- Crawlen basierend auf Ereignissen: Verlassen Sie sich auf Amazon-S3-Ereignisse, um zu steuern, welche Ordner gecrawlt werden sollen. Weitere Informationen finden Sie unter Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen.

Benutzerdefinierte Classifier – optional

Definieren Sie benutzerdefinierte Classifier, bevor Sie Crawler definieren. Ein Classifier prüft, ob eine bestimmte Datei in einem Format vorliegt, das der Crawler verarbeiten kann. Wenn dies der Fall ist, erstellt der Classifier ein Schema in Form eines `StructType`-Objekts, das dem Datenformat entspricht.

Weitere Informationen finden Sie unter Klassifikatoren definieren und verwalten.

Schritt 3: Konfigurieren der Sicherheitseinstellungen

IAM-Rolle

Der Crawler übernimmt diese Rolle. Sie muss über ähnliche Berechtigungen wie die AWS verwaltete Richtlinie verfügen`AWSGlueServiceRole`. Für Amazon-S3- und DynamoDB-Quellen muss er auch über Berechtigungen für den Zugriff auf den Datenspeicher verfügen. Wenn der Crawler mit AWS Key Management Service (AWS KMS) verschlüsselte Amazon S3 S3-Daten liest, muss die Rolle über Entschlüsselungsberechtigungen für den AWS KMS Schlüssel verfügen.

Für einen Amazon-S3-Datenspeicher wären zusätzliche Berechtigungen, die der Rolle zugeordnet sind, ähnlich wie die folgenden:

JSON

```
{
   "Version": "2012-10-17",
    "Statement": [
```

```
        {
          "Effect": "Allow",
          "Action": [
              "s3:GetObject",
              "s3:PutObject"
          ],
          "Resource": [
              "arn:aws:s3:::bucket/object*"
          ]
        }
    ]
}
```

Für einen Amazon-DynamoDB-Datenspeicher wären zusätzliche Berechtigungen, die der Rolle zugeordnet sind, ähnlich wie die folgenden:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:region:account-id:table/table-name*"
      ]
    }
  ]
}
```

Um Ihren eigenen JDBC-Treiber hinzuzufügen, müssen zusätzliche Berechtigungen hinzugefügt werden.

- Gewähren Sie Berechtigungen für die folgenden Auftragsaktionen: `CreateJob`, `DeleteJob`, `GetJob`, `GetJobRun`, `StartJobRun`.
- Gewähren Sie Berechtigungen für alle Amazon-S3-Aktionen: `s3:DeleteObjects`, `s3:GetObject`, `s3:ListBucket`, `s3:PutObject`.

> **Note**
>
> Der `s3:ListBucket` ist nicht erforderlich, wenn die Amazon-S3-Bucket-Richtlinie deaktiviert ist.

- Gewähren Sie Service Principal Zugriff auf bucket/folder in der Amazon S3 S3-Richtlinie.

Beispiel einer Amazon-S3-Richtlinie:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket-name/driver-parent-folder/driver.jar",
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

AWS Glue erstellt die folgenden Ordner (`_crawler` und `_glue_job_crawler` auf derselben Ebene wie der JDBC-Treiber in Ihrem Amazon-S3-Bucket. Wenn der Treiberpfad beispielsweise `<s3-path/driver_folder/driver.jar>` lautet, werden die folgenden Ordner erstellt, sofern sie noch nicht vorhanden sind:

- <s3-path/driver_folder/_crawler>
- <s3-path/driver_folder/_glue_job_crawler>

Optional können Sie einem Crawler eine Sicherheitskonfiguration hinzufügen, um Verschlüsselungsoptionen im Ruhezustand festzulegen.

Weitere Informationen erhalten Sie unter Schritt 2: Erstellen einer IAM-Rolle für AWS Glue und Identitäts- und Zugriffsmanagement für AWS Glue.

Lake-Formation-Konfiguration – optional

Erlauben Sie dem Crawler, Lake-Formation-Anmeldeinformationen für das Crawling der Datenquelle zu verwenden.

Wenn Sie Use Lake Formation credentials for crawling S3 data source (Lake-Formation-Anmeldeinformationen für das Crawling der S3-Datenquelle verwenden) aktivieren, kann der Crawler Lake-Formation-Anmeldeinformationen für das Crawling der Datenquelle verwenden. Wenn die Datenquelle zu einem anderen Konto gehört, müssen Sie die registrierte Konto-ID angeben. Andernfalls crawlt der Crawler nur die Datenquellen, die dem Konto zugeordnet sind. Gilt nur für Amazon-S3- und Data-Catalog-Datenquellen.

Sicherheitskonfiguration – optional

Die Einstellungen enthalten Sicherheitskonfigurationen. Weitere Informationen finden Sie hier:

- Verschlüsseln von Daten, die von AWS Glue geschrieben werden

> **Note**
>
> Sobald eine Sicherheitskonfiguration für einen Crawler eingerichtet wurde, können Sie sie ändern, aber nicht entfernen. Um die Sicherheitsstufe eines Crawlers zu verringern, legen Sie die Sicherheitsfunktion explizit `DISABLED` in Ihrer Konfiguration fest oder erstellen Sie einen neuen Crawler.

Schritt 4: Festlegen von Ausgabe und Zeitplanung

Konfiguration der Ausgabe

Zu den Optionen gehören, wie der Crawler u. a. erkannte Schemaänderungen und gelöschte Objekte im Datenspeicher verarbeiten sollte. Weitere Informationen finden Sie unter Crawler-Verhalten anpassen

Crawler-Zeitplan

Sie können einen Crawler bei Bedarf ausführen oder einen zeitbasierten Zeitplan für Ihre Crawler und Aufträge in AWS Glue definieren. Die Definition dieser Zeitpläne verwendet die Unix-ähnliche Cron-Syntax. Weitere Informationen finden Sie unter Planen eines Crawlers.

Schritt 5: Überprüfen und Erstellen

Überprüfen Sie die von Ihnen konfigurierten Crawler-Einstellungen und erstellen Sie den Crawler.

Planen eines Crawlers

Sie können eine ausführen AWS Glue Crawler auf Abruf oder nach einem regelmäßigen Zeitplan. Wenn Sie einen Crawler auf der Grundlage eines Zeitplans einrichten, können Sie bestimmte Einschränkungen angeben, z. B. die Häufigkeit, mit der der Crawler ausgeführt wird, an welchen Wochentagen und zu welcher Uhrzeit er ausgeführt wird. Sie können diese benutzerdefinierten Zeitpläne im Cron-Format erstellen. Weitere Informationen finden Sie unter Cron in Wikipedia.

Wenn Sie einen Crawler-Zeitplan erstellen, sollten Sie die Features und Einschränkungen von Cron berücksichtigen. Wenn Sie z. B. Ihren Crawler jeden Monat am 31. ausführen möchten, denken Sie daran, dass einige Monate keine 31 Tage haben.

Themen

- Erstellen Sie einen Crawler-Zeitplan
- Erstellen Sie einen Zeitplan für einen vorhandenen Crawler

Erstellen Sie einen Crawler-Zeitplan

Sie können einen Zeitplan für den Crawler mithilfe der AWS Glue Konsole oder erstellen. AWS CLI

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Crawlers aus.
3. Folgen Sie den Schritten 1—3 im Konfiguration eines Crawlers Abschnitt.
4. Wählen Sie Schritt 4: Festlegen von Ausgabe und Zeitplanung unter einen Crawler-Zeitplan aus, um die Häufigkeit der Ausführung festzulegen. Sie können den Crawler so

wählen, dass er stündlich, täglich, wöchentlich oder monatlich ausgeführt wird, oder einen benutzerdefinierten Zeitplan mithilfe von Cron-Ausdrücken definieren.

Ein Cron-Ausdruck ist eine Zeichenfolge, die ein Zeitplanmuster darstellt und aus 6 durch Leerzeichen getrennten Feldern besteht: * * * * * <minute><hour><day of month><month><day of week><year>

Um beispielsweise eine Aufgabe jeden Tag um Mitternacht auszuführen, lautet der Cron-Ausdruck: 0 0 * *? *

Weitere Informationen finden Sie unter Cron-Ausdrücke.

5. Überprüfen Sie die von Ihnen konfigurierten Crawler-Einstellungen und erstellen Sie den Crawler so, dass er nach einem Zeitplan ausgeführt wird.

AWS CLI

```
aws glue create-crawler
 --name myCrawler \
 --role AWSGlueServiceRole-myCrawler  \
 --targets '{"S3Targets":[{Path="s3://amzn-s3-demo-bucket/"}]}' \
 --schedule cron(15 12 * * ? *)
```

Weitere Informationen über die Verwendung von Cron zum Planen von Aufträgen und Crawlern finden Sie unter Zeitpläne für Aufträge und Crawler.

Erstellen Sie einen Zeitplan für einen vorhandenen Crawler

Gehen Sie wie folgt vor, um einen wiederkehrenden Zeitplan für einen vorhandenen Crawler einzurichten.

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Crawlers aus.
3. Wählen Sie aus der verfügbaren Liste einen Crawler aus, den Sie planen möchten.
4. Wählen Sie im Menü Aktionen die Option Bearbeiten aus.

5. Scrollen Sie nach unten zu Schritt 4: Ausgabe und Planung festlegen und wählen Sie „Bearbeiten“.
6. Aktualisieren Sie Ihren Crawler-Zeitplan unter Crawler-Zeitplan.
7. Wählen Sie Aktualisieren.

AWS CLI

Verwenden Sie den folgenden CLI-Befehl, um eine bestehende Crawler-Konfiguration zu aktualisieren:

```
aws glue update-crawler-schedule
   --crawler-name myCrawler
   --schedule cron(15 12 * * ? *)
```

Anzeigen von Crawler-Ergebnissen und -Details

Nachdem der Crawler erfolgreich ausgeführt wurde, erstellt er Tabellendefinitionen im Data Catalog. Wählen Sie im Navigationsbereich die Option Tables (Tabellen) aus, um die Tabellen anzuzeigen, die von Ihrem Crawler in der von Ihnen angegebenen Datenbank erstellt wurden.

Sie können Informationen zum Crawler selbst wie folgt anzeigen:

- Auf der Crawler-Seite in der AWS Glue Konsole werden die folgenden Eigenschaften für einen Crawler angezeigt:

Property (Eigenschaft)	Description (Beschreibung)
Name	Wenn Sie einen Crawler erstellen, müssen Sie ihm einen eindeutigen Namen geben.
Status	Ein Crawler-Status kann Bereit, Starten, Stoppen, Geplant oder Zeitplan angehalte n sein. Ein laufender Crawler schreitet von Starten bis Stoppen fort. Sie können einen Zeitplan, der einem Crawler angefügt ist, fortsetzen oder anhalten.

Property (Eigenschaft)	Description (Beschreibung)
Plan	Sie können Ihren Crawler nach Bedarf ausführen oder eine Häufigkeit nach einem Zeitplan auswählen. Weitere Informationen zur Planung eines Crawlers finden Sie unter Planen eines Crawlers.
Letzte Ausführung	Das Datum und die Uhrzeit der letzten Ausführung des Crawler.
Protokoll	Links zu allen verfügbaren Protokolle aus der letzten Ausführung des Crawlers.
Tabellenänderungen gegenüber der letzten Ausführung	Die Anzahl der Tabellen in der AWS Glue Data Catalog , die bei der letzten Ausführung des Crawlers aktualisiert wurden.

- Um den Verlauf für einen Crawler anzuzeigen, wählen Sie Crawler im Navigationsbereich, um die von Ihnen erstellten Crawler anzuzeigen. Wählen Sie einen Crawler aus der Liste der verfügbaren Crawler aus. Sie können die Crawler-Eigenschaften und den Crawler-Verlauf auf der Registerkarte Crawler runs (Crawler-Ausführungen) einsehen.

  Auf der Registerkarte „Crawler runs“ (Crawler-Ausführungen) werden Informationen zu jedem Zeitpunkt angezeigt, zu dem der Crawler ausgeführt wurde, einschließlich Startzeit (UTC), Endzeit (UTC),Dauer, Status, DPU-Stunden und Tabellenänderungen.

  Die Registerkarte der Crawler-Ausführungen gibt nur die Crawls zurück, die seit dem Startdatum des Crawler-Verlaufs-Features stattgefunden haben, und behält Crawls nur bis zu 12 Monate bei. Ältere Crawls werden nicht zurückgegeben.
- Um zusätzliche Informationen anzuzeigen, wählen Sie eine Registerkarte auf der Seite mit den Crawler-Details aus. Auf jeder Registerkarte werden Informationen zum Crawler angezeigt.
  - Plan: Alle für den Crawler erstellten Zeitpläne werden hier angezeigt.
  - Datenquellen: Alle vom Crawler gescannten Datenquellen werden hier angezeigt.
  - Classifier: Alle dem Crawler zugewiesenen Classifier werden hier angezeigt.
  - Tags: Alle Tags, die erstellt und einer AWS Ressource zugewiesen wurden, werden hier angezeigt.

Parameter, die vom Crawler in Data-Catalog-Tabellen festgelegt wurden

Diese Tabelleneigenschaften werden von AWS Glue Crawlern festgelegt. Wir erwarten, dass Benutzer die `classification`- und `compressionType`-Eigenschaften nutzen. Andere Eigenschaften, einschließlich Schätzungen der Tabellengröße, werden für interne Berechnungen verwendet, und wir übernehmen keine Garantie für deren Richtigkeit oder Anwendbarkeit auf Kundenanwendungsfälle. Das Ändern dieser Parameter kann das Verhalten des Crawlers verändern. Wir unterstützen diesen Workflow nicht.

Eigenschaftsschlüssel	Eigenschaftenwert
`UPDATED_BY_CRAWLER`	Name des Crawlers, der die Aktualisierung durchführt.
`connectionName`	Der Name der Verbindung im Data Catalog für den Crawler, der zum Herstellen einer Verbindung mit dem Datenspeicher verwendet wird.
`recordCount`	Schätzen Sie die Anzahl der Datensätze in der Tabelle, basierend auf Dateigrößen und Kopfzeilen.
`skip.header.line.count`	Zeilen wurden übersprungen, um die Kopfzeile zu überspringen. Wird auf Tabellen gesetzt, die als CSV klassifiziert sind.
`CrawlerSchemaSerializerVersion`	Zur internen Verwendung
`classification`	Format der Daten, abgeleitet vom Crawler. Weitere Informationen zu den von AWS Glue Crawlern unterstützten Datenformaten finden Sie unter. the section called “Integrierte Klassifikatoren”
`CrawlerSchemaDeserializerVersion`	Zur internen Verwendung
`sizeKey`	Kombinierte Größe der Dateien in der gecrawlten Tabelle.
`averageRecordSize`	Durchschnittliche Zeilengröße in der Tabelle, in Bytes.

Eigenschaftsschlüssel	Eigenschaftenwert
`compressionType`	Art der Komprimierung, die für Daten in der Tabelle verwendet wird. Weitere Informationen zu den von AWS Glue Crawlern unterstützten Komprimierungstypen finden Sie unter. the section called “Integrierte Klassifikatoren”
`typeOfData`	`file`, `table` oder `view`.
`objectCount`	Anzahl der Objekte unter dem Amazon-S3-Pfad für die Tabelle.

Diese zusätzlichen Tabelleneigenschaften werden von AWS Glue Crawlern für Snowflake-Datenspeicher festgelegt.

Eigenschaftsschlüssel	Eigenschaftenwert
`aws:RawTableLastAltered`	Zeichnet den letzten geänderten Zeitstempel der Snowflake-Tabelle auf.
`ViewOriginalText`	Zeigt die SQL-Anweisung an.
`ViewExpandedText`	Zeigt eine im Base64-Format codierte SQL-Anweisung an.
`ExternalTable:S3Location`	Amazon-S3-Speicherort der externen Snowflake-Tabelle.
`ExternalTable:FileFormat`	Amazon S3-Dateiformat der externen Snowflake-Tabelle.

Diese zusätzlichen Tabelleneigenschaften werden von AWS Glue Crawlern für Datenspeicher vom Typ JDBC wie Amazon Redshift, Microsoft SQL Server, MySQL, PostgreSQL und Oracle festgelegt.

Eigenschaftsschlüssel	Eigenschaftenwert
`aws:RawType`	Wenn ein Crawler die Daten im Data Catalog speichert, übersetzt er die Datentypen in Hive-kompatible Typen, was häufig dazu führt, dass die Informationen zum nativen Datentyp verloren gehen. Der Crawler gibt den `aws:RawType` -Parameter aus, um den Datentyp auf nativer Ebene bereitzustellen.
`aws:RawColumnComment`	Wenn einer Spalte in der Datenbank ein Kommentar zugeordnet ist, gibt der Crawler den entsprechenden Kommentar in der Katalogtabelle aus. Die Kommentarzeichenfolge wird auf 255 Byte gekürzt. Kommentare werden für Microsoft SQL Server nicht unterstützt.
`aws:RawTableComment`	Wenn einer Tabelle in der Datenbank ein Kommentar zugeordnet ist, gibt der Crawler den entsprechenden Kommentar in der Katalogtabelle aus. Die Kommentarzeichenfolge wird auf 255 Byte gekürzt. Kommentare werden für Microsoft SQL Server nicht unterstützt.

Crawler-Verhalten anpassen

Wenn Sie einen konfigurieren AWS-Glue-Crawler, haben Sie mehrere Möglichkeiten, das Verhalten Ihres Crawlers zu definieren.

- Inkrementelle Crawls — Sie können einen Crawler so konfigurieren, dass er inkrementelle Crawls ausführt, um dem Tabellenschema nur neue Partitionen hinzuzufügen.
- Partitionsindizes — Ein Crawler erstellt standardmäßig Partitionsindizes für Amazon S3- und Delta Lake-Ziele, um eine effiziente Suche nach bestimmten Partitionen zu ermöglichen.
- Beschleunigen Sie die Crawlzeit mithilfe von Amazon S3-Ereignissen — Sie können einen Crawler so konfigurieren, dass er Amazon S3 S3-Ereignisse verwendet, um die Änderungen zwischen zwei Crawls zu identifizieren, indem Sie alle Dateien aus dem Unterordner auflisten, der das Ereignis ausgelöst hat, anstatt das vollständige Amazon S3- oder Data Catalog-Ziel aufzulisten.
- Umgang mit Schemaänderungen — Sie können verhindern, dass ein Crawler Schemaänderungen am vorhandenen Schema vornimmt. Sie können das AWS Management Console oder das verwenden AWS Glue API, um zu konfigurieren, wie Ihr Crawler bestimmte Arten von Änderungen verarbeitet.

- Ein einziges Schema für mehrere Amazon S3 S3-Pfade — Sie können einen Crawler so konfigurieren, dass er für jeden S3-Pfad ein einzelnes Schema erstellt, sofern die Daten kompatibel sind.
- Tabellenposition und Partitionierungsebenen — Die Crawler-Option auf Tabellenebene bietet Ihnen die Flexibilität, dem Crawler mitzuteilen, wo sich die Tabellen befinden und wie Partitionen erstellt werden sollen.
- Tabellenschwellenwert — Sie können die maximale Anzahl von Tabellen angeben, die der Crawler erstellen darf, indem Sie einen Tabellenschwellenwert angeben.
- AWS Lake Formation Anmeldeinformationen — Sie können einen Crawler so konfigurieren, dass er Lake Formation Formation-Anmeldeinformationen verwendet, um auf einen Amazon S3 S3-Datenspeicher oder eine Datenkatalogtabelle mit einem zugrunde liegenden Amazon S3 S3-Speicherort innerhalb desselben AWS-Konto oder eines anderen AWS-Konto zuzugreifen.

Weitere Informationen zur Verwendung von AWS Glue Konsole zum Hinzufügen eines Crawlers finden Sie unterKonfiguration eines Crawlers.

Themen

- Planung inkrementeller Crawls zum Hinzufügen neuer Partitionen
- Generieren von Partitionsindizes
- Verhindern, dass ein Crawler ein vorhandenes Schema ändert
- Erstellen eines einzelnen Schemas für jeden Amazon S3 S3-Include-Pfad
- Angabe des Tabellenspeicherorts und der Partitionierungsebene
- Angabe der maximalen Anzahl von Tabellen, die der Crawler erstellen darf
- Konfiguration eines Crawlers für die Verwendung von Lake Formation Formation-Anmeldeinformationen
- Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen

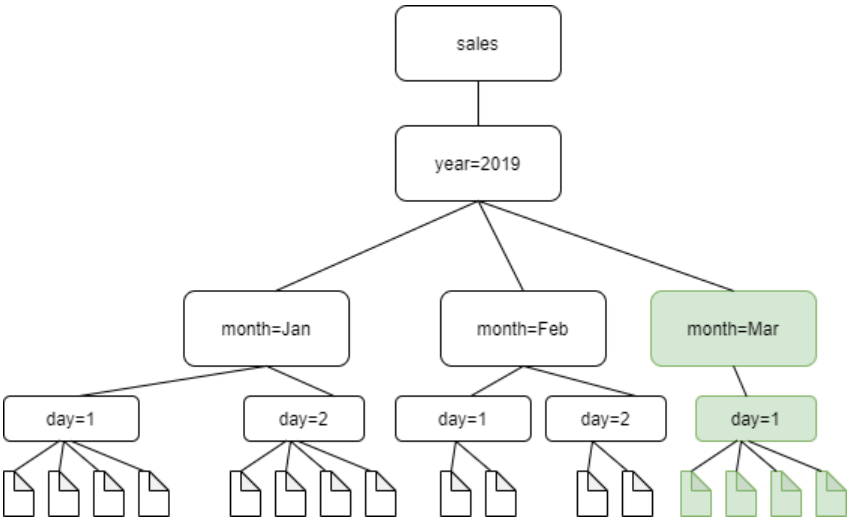
Planung inkrementeller Crawls zum Hinzufügen neuer Partitionen

Sie können inkrementelle Crawls so konfigurieren und AWS-Glue-Crawler ausführen, dass dem Tabellenschema nur neue Partitionen hinzugefügt werden. Wenn der Crawler zum ersten Mal ausgeführt wird, führt er einen vollständigen Crawl durch, um die gesamte Datenquelle zu verarbeiten und das vollständige Schema und alle vorhandenen Partitionen in der aufzuzeichnen. AWS Glue Data Catalog

Nachfolgende Durchforstungen nach dem ersten vollständigen Crawl erfolgen inkrementell, wobei der Crawler nur die neuen Partitionen identifiziert und hinzufügt, die seit dem vorherigen Crawl eingeführt wurden. Dieser Ansatz führt zu schnelleren Crawlzeiten, da der Crawler nicht mehr die gesamte Datenquelle für jeden Lauf verarbeiten muss, sondern sich stattdessen nur auf die neuen Partitionen konzentriert.

> **Note**
>
> Inkrementelle Crawls erkennen keine Änderungen oder Löschungen vorhandener Partitionen. Diese Konfiguration eignet sich am besten für Datenquellen mit einem stabilen Schema. Wenn eine einmalige größere Schemaänderung stattfindet, ist es ratsam, den Crawler vorübergehend so einzustellen, dass er einen vollständigen Crawl durchführt, um das neue Schema korrekt zu erfassen, und dann wieder in den inkrementellen Crawling-Modus zu wechseln.

Das folgende Diagramm zeigt, dass der Crawler bei aktivierter Einstellung für den inkrementellen Crawl nur den neu hinzugefügten Ordner, month=March, erkennt und dem Katalog hinzufügt.



Gehen Sie wie folgt vor, um Ihren Crawler so zu aktualisieren, dass er inkrementelle Crawls durchführt:

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/
2. Wählen Sie unter dem Datenkatalog die Option Crawlers aus.

3. Wählen Sie einen Crawler aus, den Sie für das inkrementelle Crawlen einrichten möchten.
4. Wählen Sie Edit (Bearbeiten) aus.
5. Wählen Sie Schritt 2. Wählen Sie Datenquellen und Klassifikatoren aus.
6. Wählen Sie die Datenquelle aus, die Sie inkrementell crawlen möchten.
7. Wählen Sie Edit (Bearbeiten) aus.
8. Wählen Sie unter Nachfolgende Crawlerläufe die Option Nur neue Unterordner crawlen aus.
9. Wählen Sie Aktualisieren.

Informationen zum Erstellen eines Zeitplans für einen Crawler finden Sie unter. the section called “Planen eines Crawlers”

AWS CLI

```
aws glue update-crawler \
 --name myCrawler \
 --recrawl-policy RecrawlBehavior=CRAWL_NEW_FOLDERS_ONLY \
 --schema-change-policy UpdateBehavior=LOG,DeleteBehavior=LOG
```

Hinweise und Einschränkungen

Wenn diese Option aktiviert ist, können Sie die Amazon-S3-Zieldatenspeicher beim Bearbeiten des Crawlers nicht ändern. Diese Option wirkt sich auf bestimmte Crawler-Konfigurationseinstellungen aus. Wenn diese Option aktiviert ist, erzwingt sie das Aktualisierungs- und Löschverhalten des Crawlers für `LOG`. Dies bedeutet, dass:

- Wenn er Objekte entdeckt, bei denen Schemas nicht kompatibel sind, fügt der Crawler die Objekte nicht dem Datenkatalog hinzu und fügt dieses Detail als Protokoll in Logs hinzu. CloudWatch
- Gelöschte Objekte werden im Datenkatalog nicht aktualisiert.

Generieren von Partitionsindizes

Der Datenkatalog unterstützt die Erstellung von Partitionsindizes, um eine effiziente Suche nach bestimmten Partitionen zu ermöglichen. Weitere Informationen finden Sie unter Partitionsindizes erstellen. Der AWS Glue Crawler erstellt standardmäßig Partitionsindizes für Amazon S3- und Delta Lake-Ziele.

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/
2. Wählen Sie unter dem Datenkatalog die Option Crawlers aus.
3. Wenn Sie einen Crawler definieren, ist die Option zum automatischen Erstellen von Partitionsindizes standardmäßig unter Erweiterte Optionen auf der Seite Ausgabe und Planung festlegen aktiviert.

   Um diese Option zu deaktivieren, können Sie das Kontrollkästchen Partitionsindizes automatisch erstellen in der Konsole deaktivieren.
4. Schließen Sie die Crawler-Konfiguration ab und wählen Sie Create Crawler aus.

AWS CLI

Sie können diese Option auch deaktivieren, indem Sie den AWS CLI, `CreatePartitionIndex` im Parameter festlegen, verwenden. `configuration` Der Standardwert ist "True".

```
aws glue update-crawler \
    --name myCrawler \
    --configuration '{"Version": 1.0, "CreatePartitionIndex": false }'
```

Nutzungshinweise für Partitionsindizes

- Vom Crawler erstellte Tabellen verfügen standardmäßig nicht über die Variable `partition_filtering.enabled`. Weitere Informationen finden Sie unter AWS Glue - Partitionsindizierung und -filterung.
- Das Erstellen von Partitionsindizes für verschlüsselte Partitionen wird nicht unterstützt.

Verhindern, dass ein Crawler ein vorhandenes Schema ändert

Sie können verhindern, dass AWS-Glue-Crawler s bei der Ausführung Schemaänderungen am Datenkatalog vornimmt. Standardmäßig aktualisieren Crawler das Schema im Datenkatalog so, dass es der Datenquelle entspricht, die gecrawlt wird. In einigen Fällen möchten Sie jedoch möglicherweise verhindern, dass der Crawler das vorhandene Schema ändert, insbesondere wenn Sie die Daten transformiert oder bereinigt haben und nicht möchten, dass das ursprüngliche Schema die Änderungen überschreibt.

Gehen Sie wie folgt vor, um Ihren Crawler so zu konfigurieren, dass er das bestehende Schema in einer Tabellendefinition nicht überschreibt.

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/
2. Wählen Sie unter dem Datenkatalog die Option Crawlers aus.
3. Wählen Sie einen Crawler aus der Liste aus und klicken Sie auf Bearbeiten.
4. Wählen Sie Schritt 4, Ausgabe und Planung festlegen.
5. Wählen Sie unter Erweiterte Optionen die Option Nur neue Spalten hinzufügen oder Änderung ignorieren und Tabelle im Datenkatalog nicht aktualisieren aus.
6. Sie können auch eine Konfigurationsoption festlegen, um alle neuen und vorhandenen Partitionen mit Metadaten aus der Tabelle zu aktualisieren. Dadurch werden Partitionsschemas so festgelegt, dass sie von der Tabelle erben werden.
7. Wählen Sie Aktualisieren.

AWS CLI

Das folgende Beispiel zeigt, wie ein Crawler so konfiguriert wird, dass er das bestehende Schema nicht ändert, sondern nur neue Spalten hinzufügt:

```
aws glue update-crawler \
  --name myCrawler \
  --configuration '{"Version": 1.0, "CrawlerOutput": {"Tables":
 {"AddOrUpdateBehavior": "MergeNewColumns"}}}'
```

Das folgende Beispiel zeigt, wie ein Crawler so konfiguriert wird, dass er das bestehende Schema nicht ändert und keine neuen Spalten hinzufügt:

```
aws glue update-crawler \
  --name myCrawler \
  --schema-change-policy UpdateBehavior=LOG \
  --configuration '{"Version": 1.0, "CrawlerOutput": {"Partitions":
 { "AddOrUpdateBehavior": "InheritFromTable" }}}'
```

API

Wenn Sie nicht möchten, dass ein Tabellenschema geändert wird, wenn ein Crawler ausgeführt wird, legen Sie die Schemaänderungsrichtlinie auf `LOG` fest.

Legen Sie folgende Parameter fest, wenn Sie den Crawler mit der API konfigurieren:

- Legen Sie das `UpdateBehavior`-Feld in der `SchemaChangePolicy`-Struktur auf `LOG` fest.
- Legen Sie das `Configuration`-Feld mit einer Zeichenfolgendarstellung des folgenden JSON-Objekts in der Crawler-API fest, beispielsweise:

```
{
   "Version": 1.0,
   "CrawlerOutput": {
      "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" }
   }
}
```

Erstellen eines einzelnen Schemas für jeden Amazon S3 S3-Include-Pfad

Standardmäßig gilt: Wenn ein Crawler Tabellen für in Amazon S3 gespeicherte Daten definiert, berücksichtigt er sowohl die Datenkompatibilität als auch die Schema-Ähnlichkeit. Zu den berücksichtigten Datenkompatibilitätsfaktoren gehört unter anderem, ob die Daten das gleiche Format haben (z. B. JSON), den gleichen Kompressionstyp (z. B. GZIP), die Struktur des Amazon-S3-Pfades und andere Datenattribute. Die Ähnlichkeit des Schemas ist ein Maß dafür, wie ähnlich die Schemas separater Amazon-S3-Objekte sind.

Um diese Option veranschaulichen, nehmen Sie an, Sie definieren einen Crawler mit dem Include-Pfad `s3://amzn-s3-demo-bucket/table1/`. Wenn der Crawler ausgeführt wird, findet er zwei JSON-Dateien mit den folgenden Eigenschaften:

- Datei 1 – `S3://amzn-s3-demo-bucket/table1/year=2017/data1.json`
- Dateiinhalt – `{“A”: 1, “B”: 2}`
- Schema – `A:int, B:int`

- Datei 2 – `S3://amzn-s3-demo-bucket/table1/year=2018/data2.json`
- Dateiinhalt – `{“C”: 3, “D”: 4}`
- Schema – `C: int, D: int`

Standardmäßig erstellt der Crawler zwei Tabellen namens `year_2017` und `year_2018`, da die Schemas nicht ausreichend ähnlich sind. Wenn jedoch die Option Create a single schema for each S3 path (Erstellen eines einzelnen Schemas für jeden S3-Pfad) ausgewählt ist, und wenn die Daten kompatibel sind, erstellt der Crawler eine Tabelle. Die Tabelle hat das Schema `A:int,B:int,C:int,D:int` und `partitionKey year:string`.

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie unter dem Datenkatalog die Option Crawlers aus.
3. Wenn Sie einen neuen Crawler konfigurieren, wählen Sie unter Ausgabe und Planung unter Erweiterte Optionen die Option Ein einzelnes Schema für jeden S3-Pfad erstellen aus.

AWS CLI

Sie können einen Crawler konfigurieren, um `CombineCompatibleSchemas` in einer gemeinsamen Tabellendefinition durchzuführen, sofern dies möglich ist. Mit dieser Option berücksichtigt der Crawler weiterhin die Datenkompatibilität, ignoriert aber die Ähnlichkeit der spezifischen Schemata bei der Auswertung von Amazon-S3-Objekten im angegebenen Include-Pfad.

Wenn Sie den Crawler mit dem konfigurieren AWS CLI, legen Sie die folgende Konfigurationsoption fest:

```
aws glue update-crawler \
   --name myCrawler \
   --configuration '{"Version": 1.0, "Grouping": {"TableGroupingPolicy":
 "CombineCompatibleSchemas" }}'
```

API

Verwenden Sie die folgende Konfigurationsoption, wenn Sie den Crawler mit der API konfigurieren:

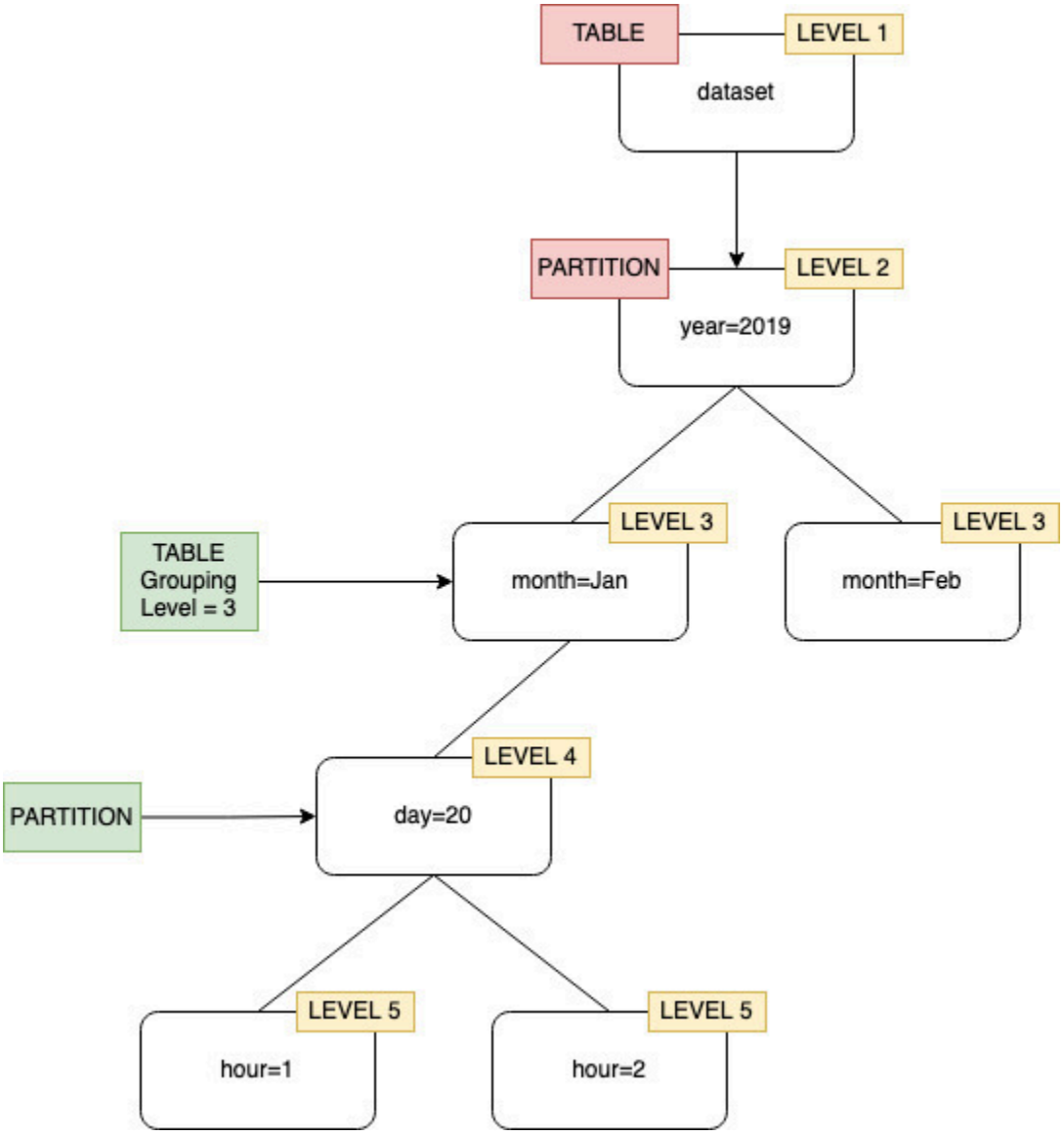
Legen Sie das `Configuration`-Feld mit einer Zeichenfolgendarstellung des folgenden JSON-Objekts in der Crawler-API fest, beispielsweise:

```
{
   "Version": 1.0,
   "Grouping": {
      "TableGroupingPolicy": "CombineCompatibleSchemas" }
}
```

Angabe des Tabellenspeicherorts und der Partitionierungsebene

Wenn ein Crawler Tabellen für in Amazon S3 gespeicherte Daten definiert, versucht der Crawler standardmäßig, Schemas zusammenzuführen und Tabellen der obersten Ebene zu erstellen (). `year=2019` In einigen Fällen kann es vorkommen, dass der Crawler eine Tabelle für den Ordner `month=Jan` erstellen soll aber stattdessen eine Partition erstellt, da ein Ordner der gleichen Ebene (`month=Mar`) in dieselbe Tabelle gespeichert wurde.

Die Crawler-Option auf Tabellenebene bietet Ihnen die Flexibilität, dem Crawler mitzuteilen, wo sich die Tabellen befinden und wie Partitionen erstellt werden sollen. Wenn Sie eine Tabellen-Ebene angeben, wird die Tabelle auf dieser absoluten Ebene aus dem Amazon S3 Bucket erstellt.

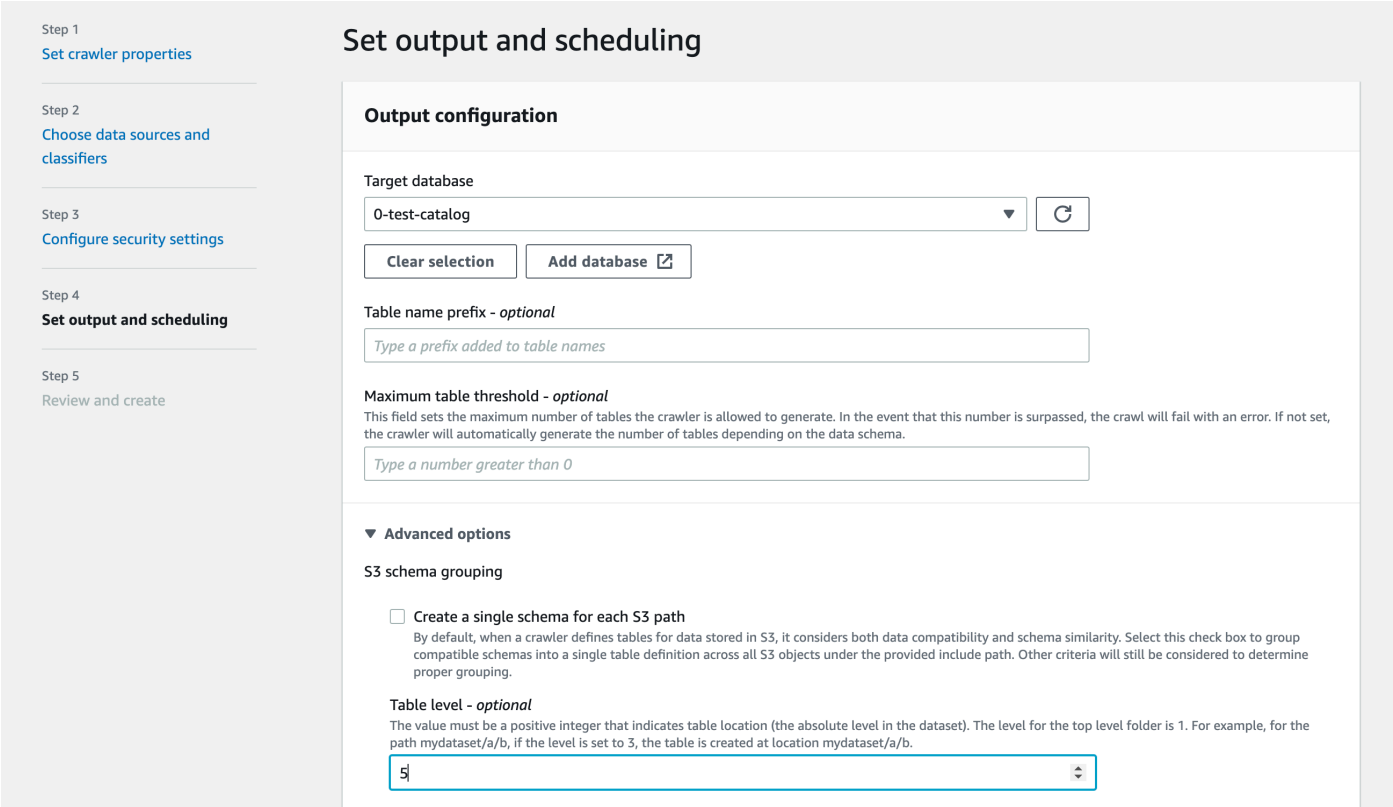


Wenn Sie den Crawler auf der Konsole konfigurieren, können Sie einen Wert für die Crawler-Option Tabellen-Ebene angeben. Der Wert muss eine positive Ganzzahl sein, die die Tabellenposition (die absolute Ebene im Datensatz) angibt. Die Ebene für den Ordner der obersten Ebene ist 1. Beispiel: Wenn die Ebene auf 3 festgelegt wurde, wird die Tabelle für den Pfad `mydataset/year/month/day/hour` am Speicherort `mydataset/year/month` erstellt.

AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Konsole unter. AWS Glue https://console.aws.amazon.com/glue/
2. Wählen Sie unter dem Datenkatalog die Option Crawlers aus.

3. Wenn Sie einen Crawler konfigurieren, wählen Sie unter Ausgabe und Planung unter Erweiterte Optionen die Option Tabellenebene aus.



AWS CLI

Wenn Sie den Crawler mit dem konfigurieren AWS CLI, legen Sie den `configuration` Parameter fest, wie im Beispielcode gezeigt:

```
aws glue update-crawler \
  --name myCrawler \
  --configuration '{"Version": 1.0, "Grouping": { "TableLevelConfiguration": 2 }}'
```

API

Legen Sie das `Configuration`-Feld mit einer Zeichenfolgendarstellung des folgenden JSON-Objekts fest, wenn Sie den Crawler mit der API konfigurieren, beispielsweise:

```
configuration = jsonencode(
{
   "Version": 1.0,
   "Grouping": {
```

```
            TableLevelConfiguration = 2
        }
})
```

CloudFormation

In diesem Beispiel legen Sie die Option auf Tabellenebene fest, die in der Konsole in Ihrer CloudFormation Vorlage verfügbar ist:

```
"Configuration": "{
    \"Version\":1.0,
    \"Grouping\":{\"TableLevelConfiguration\":2}
}"
```

Angabe der maximalen Anzahl von Tabellen, die der Crawler erstellen darf

Sie können optional die maximale Anzahl von Tabellen angeben, die der Crawler erstellen darf, indem Sie a `TableThreshold` über die AWS Glue Konsole oder angeben. AWS CLI Wenn die vom Crawler während des Crawlings erkannten Tabellen größer als dieser Eingabewert sind, schlägt das Crawling fehl und es werden keine Daten in den Data Catalog geschrieben.

Dieser Parameter ist nützlich, wenn die Tabellen, die vom Crawler erkannt und erstellt werden, viel größer sind als erwartet. Dafür kann es mehrere Gründe geben, wie zum Beispiel:

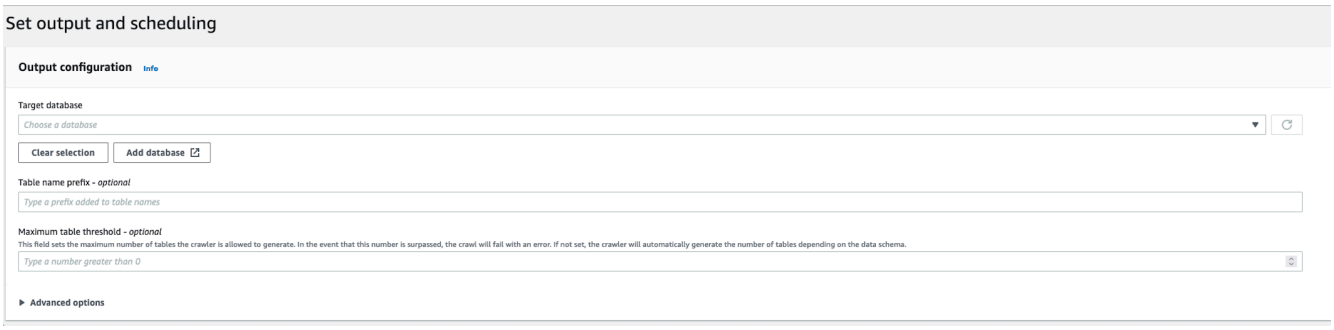
- Wenn Sie einen AWS Glue Job verwenden, um Ihre Amazon S3 S3-Speicherorte zu füllen, können Sie am Ende leere Dateien auf derselben Ebene wie ein Ordner haben. Wenn Sie in solchen Fällen einen Crawler an diesem Amazon-S3-Speicherort ausführen, erstellt der Crawler aufgrund von Dateien und Ordnern, die auf derselben Ebene vorhanden sind, mehrere Tabellen.
- Wenn Sie `"TableGroupingPolicy": "CombineCompatibleSchemas"` nicht konfigurieren, erhalten Sie möglicherweise mehr Tabellen als erwartet.

Sie geben den `TableThreshold` als Ganzzahl größer 0 an. Dieser Wert wird pro Crawler konfiguriert. Das heißt, für jedes Crawling wird dieser Wert berücksichtigt. Beispiel: Für einen Crawler wurde der `TableThreshold`-Wert auf 5 gesetzt. Bei jedem Crawl wird die Anzahl der erkannten Tabellen mit diesem Tabellen-Schwellenwert (5) AWS Glue verglichen. Wenn die Anzahl der erkannten Tabellen unter 5 liegt, werden die Tabellen in den Datenkatalog AWS Glue geschrieben. Falls nicht, schlägt der Crawl fehl, ohne in den Datenkatalog zu schreiben.

AWS Management Console

Zur Einstellung **`TableThreshold`** verwenden Sie: AWS Management Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wenn Sie einen Crawler konfigurieren, legen Sie unter Ausgabe und Planung den Schwellenwert für maximale Tabelle auf die Anzahl der Tabellen fest, die der Crawler generieren darf.



AWS CLI

Zum Einstellen `TableThreshold` mit dem AWS CLI:

```
aws glue update-crawler \
    --name myCrawler \
    --configuration '{"Version": 1.0, "CrawlerOutput": {"Tables":
 { "TableThreshold": 5 }}}'
```

API

So legen Sie es `TableThreshold` mithilfe der API fest:

```
"{"Version":1.0,
"CrawlerOutput":
{"Tables":{"AddOrUpdateBehavior":"MergeNewColumns",
"TableThreshold":5}}}";
```

Fehlermeldungen werden protokolliert, um Ihnen zu helfen, Tabellenpfade zu identifizieren und Ihre Daten zu bereinigen. Beispiel: Melden Sie sich bei Ihrem Konto an, wenn der Crawler fehlschlägt, weil die Tabellenanzahl größer als der angegebene Tabellenschwellenwert war:

```
Table Threshold value = 28, Tables detected - 29
```

Darin CloudWatch protokollieren wir alle erkannten Tabellenpositionen als INFO-Meldung. Ein Fehler wird als Grund für den Fehlschlag protokolliert.

```
ERROR com.amazonaws.services.glue.customerLogs.CustomerLogService - CustomerLogService
 received CustomerFacingException with message
The number of tables detected by crawler: 29 is greater than the table threshold value
 provided: 28. Failing crawler without writing to Data Catalog.
com.amazonaws.services.glue.exceptions.CustomerFacingInternalException: The number of
 tables detected by crawler: 29 is greater than the table threshold value provided:
 28.
Failing crawler without writing to Data Catalog.
```

Konfiguration eines Crawlers für die Verwendung von Lake Formation Formation-Anmeldeinformationen

Sie können einen Crawler so konfigurieren, dass er AWS Lake Formation Anmeldeinformationen für den Zugriff auf einen Amazon S3 S3-Datenspeicher oder eine Datenkatalogtabelle mit einem zugrunde liegenden Amazon S3 S3-Speicherort innerhalb desselben AWS-Konto oder eines anderen AWS-Konto verwendet. Sie können eine vorhandene Data-Catalog-Tabelle als Crawler-Ziel konfigurieren, wenn sich der Crawler und die Data-Catalog-Tabelle im selben Konto befinden. Derzeit ist nur ein einzelnes Katalogziel mit einer einzigen Katalogtabelle zulässig, wenn eine Datenkatalogtabelle als Ziel eines Crawlers verwendet wird.

> **Note**
>
> Wenn Sie eine Data-Catalog-Tabelle als Crawler-Ziel definieren, stellen Sie sicher, dass der zugrunde liegende Speicherort der Data-Catalog-Tabelle ein Amazon-S3-Speicherort ist. Crawler, die Lake-Formation-Anmeldeinformationen verwenden, unterstützen nur Data-Catalog-Ziele mit zugrunde liegenden Amazon-S3-Speicherorten.

Einrichtung erforderlich, wenn sich der Crawler und der registrierte Amazon-S3-Speicherort oder die Data-Catalog-Tabelle im selben Konto befinden (In-Account-Crawling)

Damit der Crawler mithilfe der Lake-Formation-Anmeldeinformationen auf einen Datenspeicher oder eine Data-Catalog-Tabelle zugreifen kann, müssen Sie den Datenspeicherort bei Lake Formation

registrieren. Außerdem muss die IAM-Rolle des Crawlers über Berechtigungen zum Lesen der Daten von dem Ziel verfügen, an dem der Amazon-S3-Bucket registriert ist.

Sie können die folgenden Konfigurationsschritte mit dem AWS Management Console oder AWS Command Line Interface (AWS CLI) ausführen.

AWS Management Console

1. Bevor Sie einen Crawler für den Zugriff auf die Crawler-Quelle konfigurieren, registrieren Sie den Datenspeicherort des Datenspeichers oder des Data Catalog bei Lake Formation. Registrieren Sie in der Lake Formation Formation-Konsole (https://console.aws.amazon.com/lakeformation/) einen Amazon S3 S3-Standort als Stammverzeichnis Ihres Data Lakes an dem Ort, AWS-Konto an dem der Crawler definiert ist. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Speicherorts.
2. Gewähren Sie der IAM-Rolle Data location (Datenspeicherort)-Berechtigungen, die für die Crawler-Ausführung verwendet wird, damit der Crawler die Daten vom Ziel in Lake Formation lesen kann. Weitere Informationen finden Sie unter Erteilen von Datenspeicherortberechtigungen (im selben Konto).
3. Gewähren Sie der Crawler-Rolle Zugriffsberechtigungen (`Create`) auf die Datenbank, die als Ausgabedatenbank angegeben ist. Weitere Informationen finden Sie unter Erteilen von Datenbankberechtigungen mit der Lake-Formation-Konsole und der benannten Ressourcenmethode.
4. Erstellen Sie in der IAM-Konsole (https://console.aws.amazon.com/iam/) eine IAM-Rolle für den Crawler. Fügen Sie die `lakeformation:GetDataAccess`-Richtlinie zur Rolle hinzu.
5. Wählen Sie in der AWS Glue Konsole (https://console.aws.amazon.com/glue/) bei der Konfiguration des Crawlers die Option Lake Formation Formation-Anmeldeinformationen für das Crawlen der Amazon S3 S3-Datenquelle verwenden aus.

> **Note**
>
> Das Feld accountId ist optional für das In-Account-Crawling.

AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
```

```
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
    "DatabaseName": "prod-run-db",
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://amzn-s3-demo-bucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111122223333"
  },
  "Configuration": {
           "Version": 1.0,
           "CrawlerOutput": {
             "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
             "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
           },
           "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
         },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

Einrichtung erforderlich, wenn sich der Crawler und der registrierte Amazon-S3-Standort im selben Konto befinden (Cross-Account-Crawling)

Damit der Crawler mit Lake-Formation-Anmeldeinformationen auf einen Datenspeicher in einem anderen Konto zugreifen kann, müssen Sie zuerst den Amazon-S3-Datenspeicherort bei Lake Formation registrieren. Anschließend erteilen Sie dem Konto des Crawlers Datenspeicherortberechtigungen, indem Sie die folgenden Schritte ausführen.

Sie können die folgenden Schritte mit dem AWS Management Console oder ausführen AWS CLI.

AWS Management Console

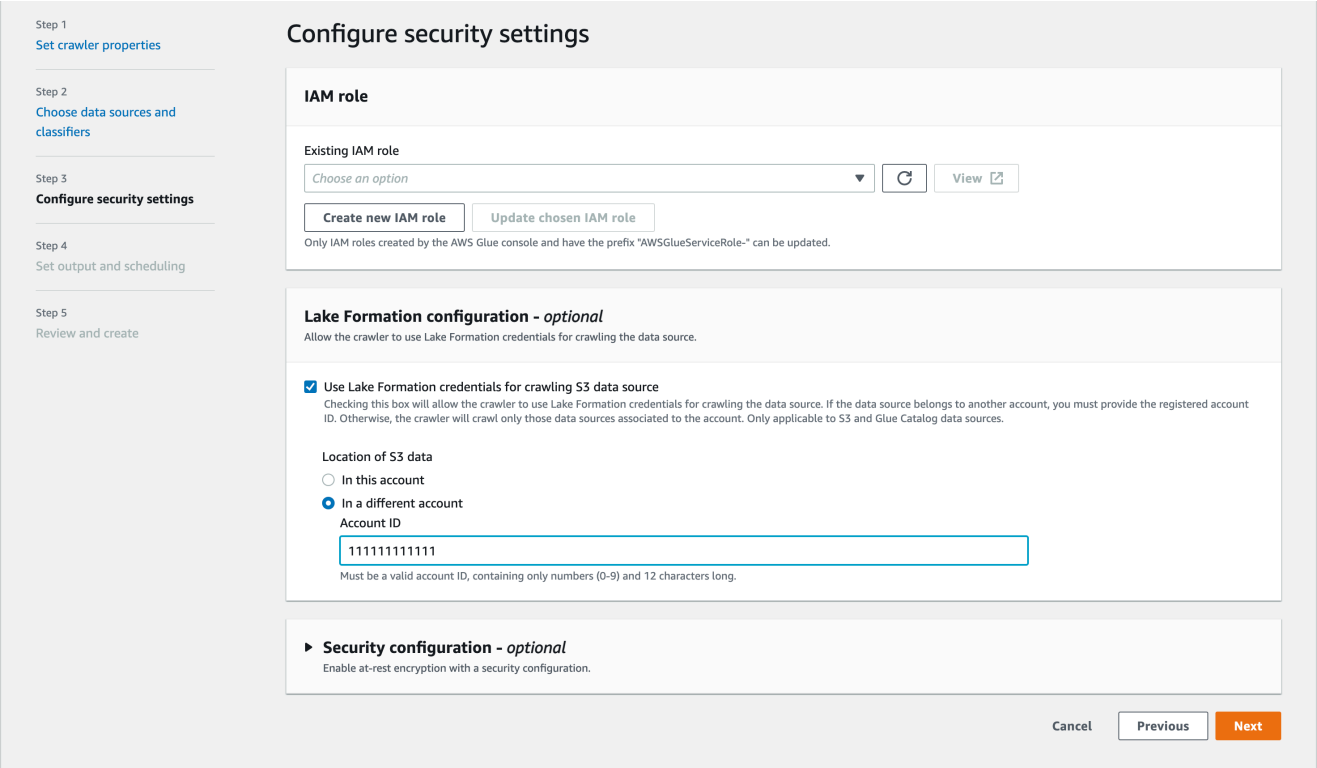
1. In dem Konto, in dem der Amazon-S3-Speicherort registriert ist (Konto B):
   a. Registrieren Sie einen Amazon-S3-Pfad bei Lake Formation. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Standorts.
   b. Gewähren Sie Data location (Datenspeicherort)-Berechtigungen für das Konto (Konto A), in dem der Crawler ausgeführt wird. Weitere Informationen finden Sie unter Erteilen von Datenstandortberechtigungen.
   c. Erstellen Sie eine leere Datenbank in Lake Formation mit dem zugrunde liegenden Speicherort als Amazon-S3-Speicherort. Weitere Informationen finden Sie unter Erstellen einer Datenbank.
   d. Gewähren Sie Konto A (das Konto, in dem der Crawler ausgeführt wird) Zugriff auf die Datenbank, die Sie im vorherigen Schritt erstellt haben. Weitere Informationen finden Sie unter Erteilen von Datenbankberechtigungen.
2. In dem Konto, in dem der Crawler erstellt und ausgeführt wird (Konto A):
   a. Akzeptieren Sie mithilfe der AWS RAM Konsole die Datenbank, die vom externen Konto (Konto B) gemeinsam genutzt wurde. Weitere Informationen finden Sie unter Annehmen einer Einladung zur gemeinsamen Nutzung einer Ressource von AWS Resource Access Manager.
   b. Erstellen Sie eine IAM-Rolle für den Crawler. Fügen Sie die `lakeformation:GetDataAccess`-Richtlinie zur Rolle hinzu.
   c. Erteilen Sie in der Lake Formation-Konsole (https://console.aws.amazon.com/lakeformation/) der IAM-Rolle, die für den Crawler-Lauf verwendet wurde, Datenstandortberechtigungen für den Amazon S3 S3-Zielstandort, damit der Crawler die Daten vom Ziel in Lake Formation lesen kann. Weitere Informationen finden Sie unter Erteilen von Datenstandortberechtigungen.

d. Erstellen Sie einen Ressourcenlink in der freigegebenen Datenbank. Weitere Informationen finden Sie unter Erstellen eines Ressourcenlinks.

e. Gewähren Sie der Crawler-Rolle Zugriffsberechtigungen (`Create`) in der gemeinsam genutzten Datenbank und (`Describe`) dem Ressourcenlink. Der Ressourcenlink wird in der Ausgabe für den Crawler angegeben.

f. Wählen Sie in der AWS Glue Konsole (https://console.aws.amazon.com/glue/) bei der Konfiguration des Crawlers die Option Lake Formation Formation-Anmeldeinformationen für das Crawlen der Amazon S3 S3-Datenquelle verwenden aus.

   Geben Sie für kontoübergreifendes Crawling die AWS-Konto ID an, unter der der Amazon S3 S3-Zielstandort bei Lake Formation registriert ist. Das Feld „accountId“ ist optional für das Crawling im Konto.



AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
    "DatabaseName": "prod-run-db",
```

```
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://amzn-s3-demo-bucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111111111111"
  },
  "Configuration": {
           "Version": 1.0,
           "CrawlerOutput": {
             "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
             "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
           },
           "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
         },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

Note

- Ein Crawler, der Lake-Formation-Anmeldeinformationen verwendet, wird nur für Amazon-S3- und Data-Catalog-Ziele unterstützt.

- Für Ziele, die Lake-Formation-Anmeldeinformationen verwenden, müssen die zugrunde liegenden Amazon-S3-Speicherorte demselben Bucket angehören. Beispielsweise ist es nicht zulässig, dass Kunden mehrere Ziele verwenden (s3://amzn-s3-demo — bucket1/folder1, s3://amzn-s3-demo-bucket1/folder2) as long as all target locations are under the same bucket (amzn-s3-demo-bucket1). Specifying different buckets (s3://amzn-s3-demo-bucket1/folder1, s3://amzn-s3-demo-bucket2/folder 2).
- Derzeit ist nur ein einzelnes Katalogziel mit einer einzigen Katalogtabelle zulässig, wenn eine Data Catalog als Ziel eines Crawlers verwendet wird.

Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen

Anstatt die Objekte aus einem Amazon S3- oder Data Catalog-Ziel aufzulisten, können Sie den Crawler so konfigurieren, dass er Amazon S3-Ereignisse verwendet, um Änderungen zu finden. Dieses Feature verbessert die Recrawl-Zeit, indem Amazon S3-Ereignisse verwendet werden, um die Änderungen zwischen zwei Crawls zu identifizieren, indem alle Dateien aus dem Unterordner aufgeführt werden, der das Ereignis ausgelöst hat, anstatt das vollständige Amazon S3- oder Data Calatog-Ziel aufzulisten.

Der erste Crawl listet alle Amazon S3-Objekte aus dem Ziel auf. Nach dem ersten erfolgreichen Crawl können Sie wählen, ob Sie manuell oder nach einem festgelegten Zeitplan recrawlen möchten. Der Crawler listet nur die Objekte dieser Ereignisse auf, anstatt alle Objekte aufzulisten.

Wenn das Ziel eine Datenkatalogtabelle ist, aktualisiert der Crawler die vorhandenen Tabellen im Datenkatalog mit Änderungen (z. B. zusätzliche Partitionen in einer Tabelle).

Die Vorteile eines Umstiegs zu einem ereignisbasierten Crawler auf Amazon S3 sind:

- Ein schnelleres Recrawl, da die Auflistung aller Objekte aus dem Ziel nicht erforderlich ist, stattdessen erfolgt die Auflistung bestimmter Ordner, in denen Objekte hinzugefügt oder gelöscht werden.
- Eine Senkung der Gesamtdurchforstungskosten, da die Auflistung bestimmter Ordner erfolgt, in denen Objekte hinzugefügt oder gelöscht werden.

Die Amazon S3-Ereignisdurchforstung wird ausgeführt, indem Amazon S3-Ereignisse aus der SQS-Warteschlange basierend auf dem Crawler-Zeitplan verwendet werden. Es fallen keine Kosten an, wenn keine Ereignisse in der Warteschlange vorhanden sind. Amazon S3-Ereignisse

können so konfiguriert werden, dass sie direkt in die SQS-Warteschlange bzw. in Fällen, in denen mehrere Verbraucher dasselbe Ereignis benötigen, in eine Kombination aus SNS und SQS gelangen. Weitere Informationen finden Sie unter the section called “Ihr Konto für Amazon S3 S3-Ereignisbenachrichtigungen einrichten”.

Nach dem Erstellen und Konfigurieren des Crawlers im Ereignismodus wird das erste Crawl im Listing-Modus ausgeführt, indem eine vollständige Auflistung des Amazon S3- oder Data Catalog-Ziels ausgeführt wird. Das folgende Protokoll bestätigt den Crawl-Vorgang durch den Verbrauch von Amazon S3-Ereignissen nach dem ersten erfolgreichen Crawl: „Der Crawl wird durch den Verbrauch von Amazon S3-Ereignissen ausgeführt.“

Nach dem Erstellen des Amazon S3-Ereigniscrawls und dem Aktualisieren der Crawler-Eigenschaften, die sich auf das Crawl auswirken können, wird das Crawl im List-Modus ausgeführt und das folgende Protokoll wird hinzugefügt: „Crawl läuft nicht im S3-Ereignismodus“.

> **Note**
>
> Die maximale Anzahl der zu verarbeitenden Nachrichten beträgt 100.000 Nachrichten pro Crawl.

Überlegungen und Einschränkungen

Die folgenden Überlegungen und Einschränkungen gelten, wenn Sie einen Crawler so konfigurieren, dass er Amazon S3 S3-Ereignisbenachrichtigungen verwendet, um Änderungen zu finden.

- Wichtiges Verhalten bei gelöschten Partitionen

  Wenn Sie Amazon S3 S3-Event-Crawler mit Datenkatalogtabellen verwenden:

  - Wenn Sie eine Partition mithilfe des `DeletePartition` API-Aufrufs löschen, müssen Sie auch alle S3-Objekte unter dieser Partition löschen und bei der Konfiguration Ihrer S3-Ereignisbenachrichtigungen die Option Alle Ereignisse zum Entfernen von Objekten auswählen. Wenn keine Löschereignisse konfiguriert sind, erstellt der Crawler die gelöschte Partition bei der nächsten Ausführung neu.
- Nur ein einziges Ziel wird vom Crawler unterstützt, unabhängig davon, ob es sich um Amazon S3- oder Data Catalog-Ziele handelt.
- SQS auf privater VPC wird nicht unterstützt.
- Amazon S3-Probenahme wird nicht unterstützt.

- Das Crawlerziel sollte ein Ordner für ein Amazon S3-Ziel oder eine oder mehrereAWS Glue Data Catalog-Tabellen für ein Data Catalog-Ziel sein.
- Der Pfad-Platzhalter „alles“ wird nicht unterstützt: s3://%
- Bei einem Data Catalog-Ziel sollten alle Katalogtabellen auf denselben Amazon S3-Bucket für den Amazon S3-Ereignismodus verweisen.
- Bei einem Data Catalog-Ziel sollte eine Katalogtabelle nicht auf einen Amazon S3-Speicherort im Delta Lake-Format verweisen (der Ordner _symlink enthält oder das `InputFormat` der Katalogtabelle überprüft).

Themen

- Ihr Konto für Amazon S3 S3-Ereignisbenachrichtigungen einrichten
- Einrichtung eines Crawlers für Amazon S3 S3-Ereignisbenachrichtigungen für ein Amazon S3 S3-Ziel
- Einrichtung eines Crawlers für Amazon S3 S3-Ereignisbenachrichtigungen für eine Datenkatalogtabelle

Ihr Konto für Amazon S3 S3-Ereignisbenachrichtigungen einrichten

Führen Sie die folgenden Einrichtungsaufgaben aus. Beachten Sie, dass die Werte in Klammern auf die konfigurierbaren Einstellungen aus dem Skript verweisen.

1. Sie müssen Ereignisbenachrichtigungen für Ihren Amazon S3 S3-Bucket einrichten.

   Weitere Informationen finden Sie unter Amazon S3 S3-Ereignisbenachrichtigungen.
2. Um den ereignisbasierten Amazon S3 S3-Crawler zu verwenden, sollten Sie die Ereignisbenachrichtigung im Amazon S3 S3-Bucket aktivieren, wobei Ereignisse aus dem Präfix gefiltert werden, das dem S3-Ziel entspricht und in SQS gespeichert ist. Sie können SQS und die Ereignisbenachrichtigung über die Konsole einrichten, indem Sie die Schritte unter Exemplarische Vorgehensweise: Konfiguration eines Buckets für Benachrichtigungen befolgen.
3. Fügen Sie der vom Crawler verwendeten Rolle die folgende SQS-Richtlinie hinzu.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:GetQueueUrl",
                "sqs:ListDeadLetterSourceQueues",
                "sqs:ReceiveMessage",
                "sqs:GetQueueAttributes",
                "sqs:ListQueueTags",
                "sqs:SetQueueAttributes",
                "sqs:PurgeQueue"
            ],
            "Resource": "arn:aws:sqs:us-east-1:{accountID}:cfn-sqs-queue"
        }
    ]
}
```

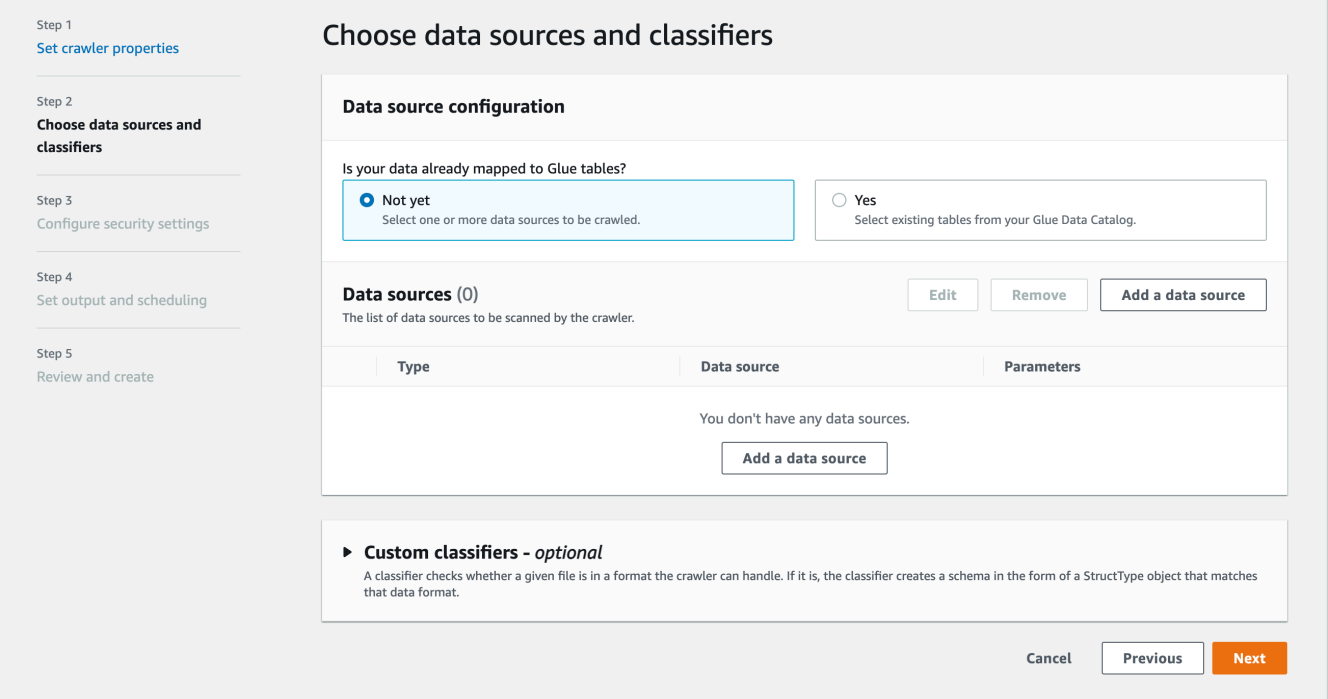
Einrichtung eines Crawlers für Amazon S3 S3-Ereignisbenachrichtigungen für ein Amazon S3 S3-Ziel

Gehen Sie wie folgt vor, um einen Crawler für Amazon S3 S3-Ereignisbenachrichtigungen für ein Amazon S3 S3-Ziel einzurichten, indem Sie die Option AWS Management Console oder AWS CLI verwenden.

AWS Management Console

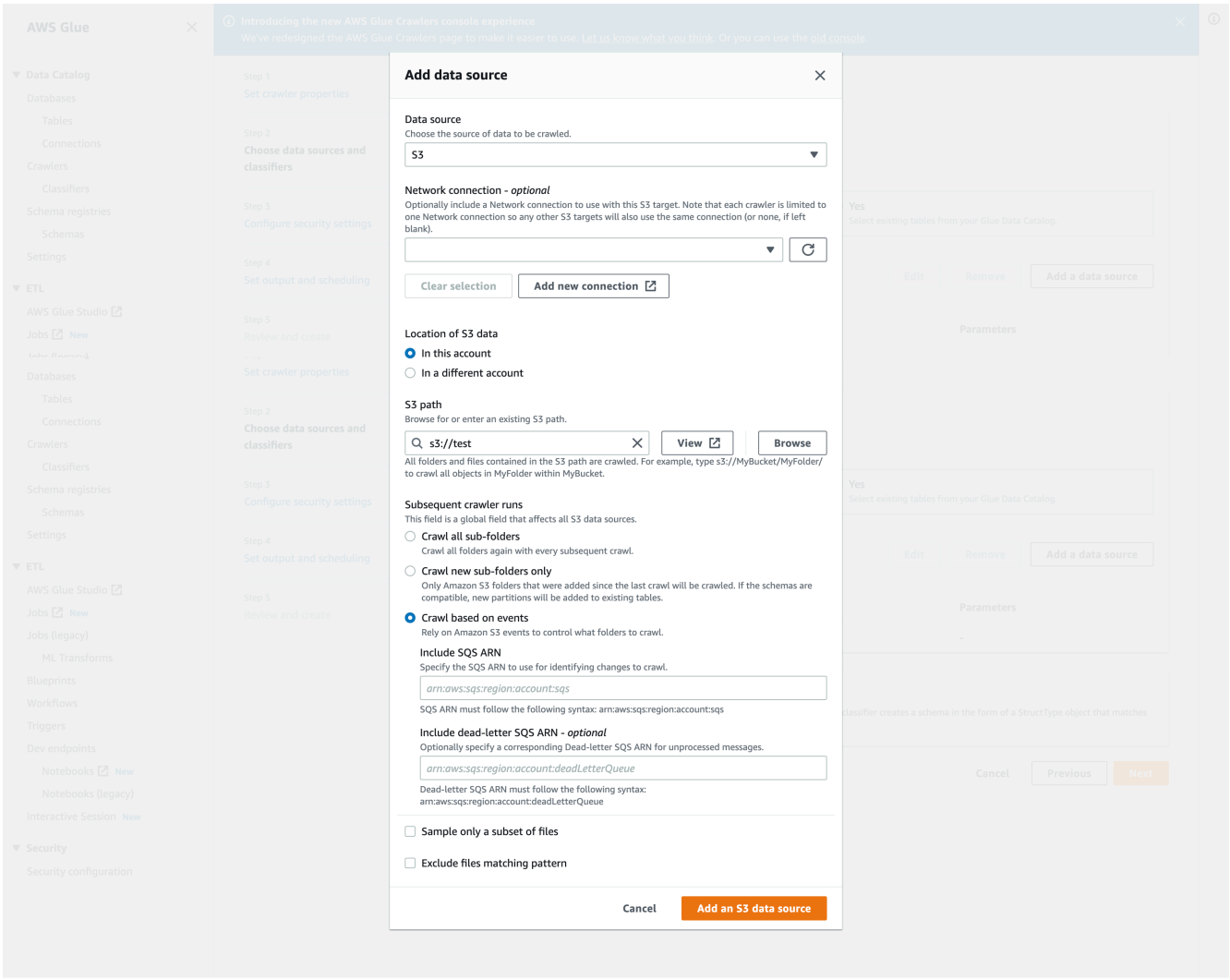
1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die GuardDuty Konsole unter https://console.aws.amazon.com/guardduty/.
2. Legen Sie Ihre Crawler-Eigenschaften fest. Weitere Informationen finden Sie unter Einstellung der Crawler-Konfigurationsoptionen auf der AWS Glue Konsole.
3. Im Abschnitt Datenquellenkonfiguration werden Sie gefragt, ob Ihre Daten bereits zugeordnet sind AWS Glue Tabellen?

   Standardmäßig ist Not yet (Noch nicht) ausgewählt. Belassen Sie dies als Standard, da Sie eine Amazon S3 S3-Datenquelle verwenden und die Daten noch nicht zugeordnet sind AWS Glue Tabellen.
4. Wählen Sie im Abschnitt Data sources (Datenquellen) Add a data source (Datenquelle hinzufügen) aus.



5. Konfigurieren Sie im Modal Add a data source (Datenquelle hinzufügen) die Amazon-S3-Datenquelle:

   - Data source (Datenquelle): Standardmäßig ist Amazon S3 ausgewählt.
   - Network connection (Netzwerkverbindung) (Optional): Wählen Sie Add new connection (Neue Verbindung hinzufügen).
   - Location of Amazon S3 data (Speicherort der Amazon-S3-Daten): Standardmäßig ist In this account (In diesem Konto) ausgewählt.
   - Amazon S3 path (Amazon-S3-Pfad): Geben Sie den Amazon-S3-Pfad an, wo Ordner und Dateien gecrawlt werden.
   - Subsequent crawler runs (Nachfolgende Crawler-Ausführungen): Wählen Sie Crawl based on events (Crawling basierend auf Ereignissen) aus, um Amazon-S3-Ereignisbenachrichtigungen für Ihren Crawler zu verwenden.
   - SQS ARN hinzufügen: Geben Sie die Datenspeicherparameter einschließlich eines gültigen SQS ARN an. (Beispiel: `arn:aws:sqs:region:account:sqs`).
   - Dead-Letter SQS ARN hinzufügen (Optional): Geben Sie einen gültigen Amazon Dead-Letter SQS ARN an. (Beispiel: `arn:aws:sqs:region:account:deadLetterQueue`).
   - Wählen Sie Add an Amazon S3 data source (Amazon-S3-Datenquelle hinzufügen) aus.



AWS CLI

Im Folgenden finden Sie ein Beispiel für einen Amazon S3 AWS CLI S3-Aufruf, um einen Crawler so zu konfigurieren, dass er Ereignisbenachrichtigungen verwendet, um einen Amazon S3 S3-Ziel-Bucket zu crawlen.

```
Create Crawler:
aws glue update-crawler \
    --name myCrawler \
    --recrawl-policy RecrawlBehavior=CRAWL_EVENT_MODE \
    --schema-change-policy UpdateBehavior=UPDATE_IN_DATABASE,DeleteBehavior=LOG
```

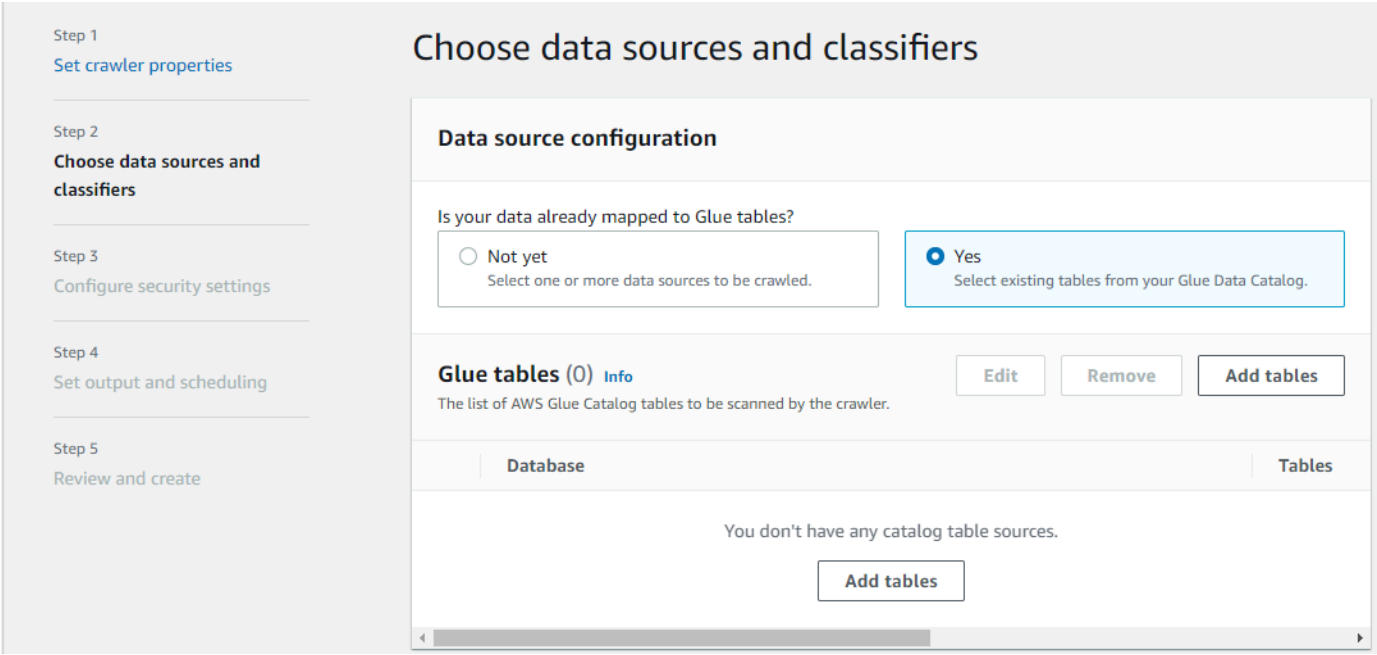
```
    --targets '{"S3Targets":[{"Path":"s3://amzn-s3-demo-bucket/", "EventQueueArn":
"arn:aws:sqs:us-east-1:012345678910:MyQueue"}]}'
```

Einrichtung eines Crawlers für Amazon S3 S3-Ereignisbenachrichtigungen für eine Datenkatalogtabelle

Wenn Sie über eine Datenkatalogtabelle verfügen, richten Sie einen Crawler für Amazon S3 S3-Ereignisbenachrichtigungen ein, indem Sie AWS Glue Konsole:

1. Legen Sie Ihre Crawler-Eigenschaften fest. Weitere Informationen finden Sie unter Einstellung der Crawler-Konfigurationsoptionen auf der AWS Glue Konsole.
2. Im Abschnitt Datenquellenkonfiguration werden Sie gefragt, ob Ihre Daten bereits zugeordnet sind AWS Glue Tabellen?

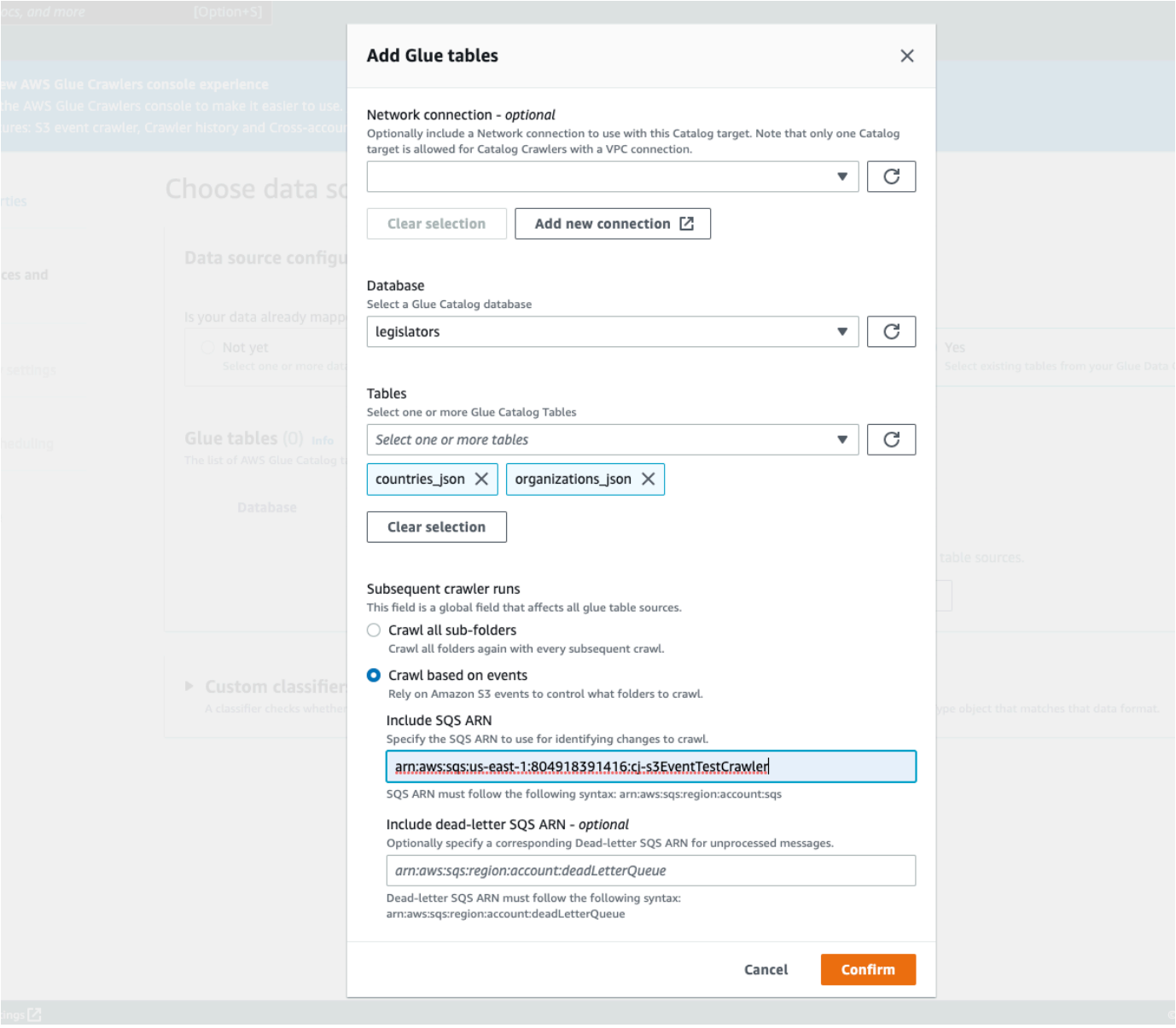
   Wählen Sie Ja, um vorhandene Tabellen aus Ihrem Data Catalog als Datenquelle auszuwählen.
3. Wählen Sie im Abschnitt Glue-Tabellen die Option Tabellen hinzufügen.



4. Konfigurieren Sie im Modal Tabelle hinzufügen die Datenbank und die Tabellen:

   - Network connection (Netzwerkverbindung) (Optional): Wählen Sie Add new connection (Neue Verbindung hinzufügen).
   - Datenbank: Wählen Sie eine Datenbank im Data Catalog.

- Tabellen: Wählen Sie eine oder mehrere Tabellen aus dieser Datenbank im Data Catalog aus.
- Subsequent crawler runs (Nachfolgende Crawler-Ausführungen): Wählen Sie Crawl based on events (Crawling basierend auf Ereignissen) aus, um Amazon-S3-Ereignisbenachrichtigungen für Ihren Crawler zu verwenden.
- SQS ARN hinzufügen: Geben Sie die Datenspeicherparameter einschließlich eines gültigen SQS ARN an. (Beispiel: `arn:aws:sqs:region:account:sqs`).
- Dead-Letter SQS ARN hinzufügen (Optional): Geben Sie einen gültigen Amazon Dead-Letter SQS ARN an. (Beispiel: `arn:aws:sqs:region:account:deadLetterQueue`).
- Wählen Sie Bestätigen aus.



Tutorial: Hinzufügen eines AWS Glue Crawler

Dafür AWS Glue In diesem Szenario werden Sie gebeten, die Ankunftsdaten großer Fluggesellschaften zu analysieren, um die Beliebtheit von Abflughäfen von Monat zu Monat zu berechnen. Sie haben Flugdaten für das Jahr 2016 im CSV-Format in Amazon S3 gespeichert. Bevor Sie Ihre Daten transformieren und analysieren, katalogisieren Sie ihre Metadaten in AWS Glue Data Catalog.

In diesem Tutorial wollen wir einen Crawler nutzen, der Metadaten aus diesen Flugprotokollen in Amazon S3 ableitet und eine Tabelle in Ihrem Data Catalog erstellt.

Themen

- Voraussetzungen
- Schritt 1: Einen Crawler erstellen
- Schritt 2: Den Crawler ausführen
- Schritt 3: Ansehen AWS Glue Data Catalog objects

Voraussetzungen

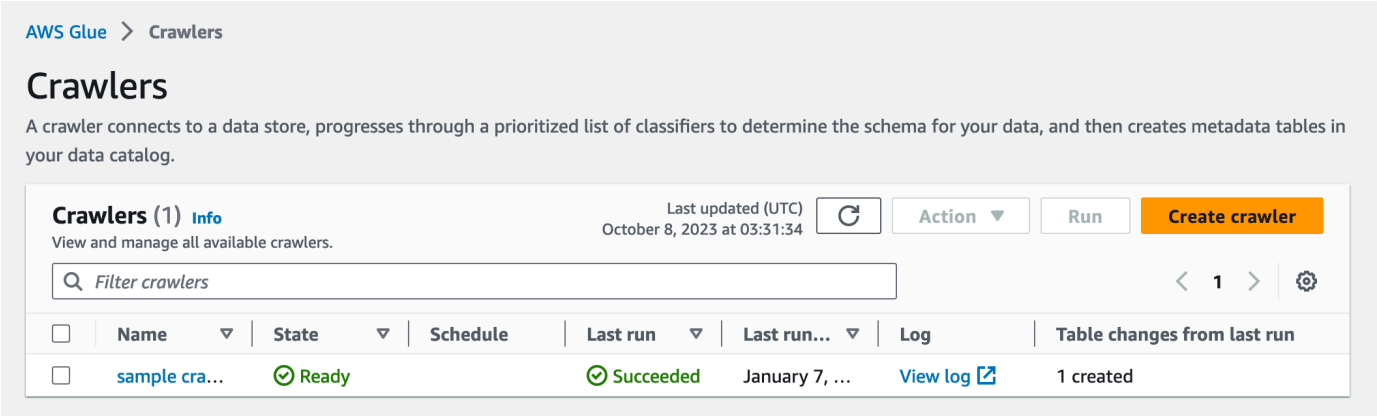
In diesem Tutorial wird davon ausgegangen, dass Sie ein AWS Konto haben und Zugriff auf AWS Glue.

Schritt 1: Einen Crawler erstellen

Die folgenden Schritte zeigen, wie Sie einen Crawler konfigurieren und ausführen, der die Metadaten aus einer CSV-Datei in Amazon S3 extrahiert.

Einen Crawler erstellen, der Dateien auf Amazon S3 liest

1. Wählen Sie in der AWS Glue Servicekonsole im Menü auf der linken Seite Crawlers aus.
2. Wählen Sie auf der Crawlers-Seite die Option Crawler erstellen aus. Es erscheint eine Folge von Seiten, auf denen Sie Details zum Crawler eingeben sollen.



3. Geben Sie im Namensfeld des Crawlers **`Flights Data Crawler`** ein und wählen Sie Next (Weiter) aus.

   Crawler rufen Classifier auf, um das Schema Ihrer Daten abzuleiten. In diesem Tutorial wird standardmäßig der integrierte Classifier für CSV verwendet.

4. Wählen Sie für den Crawler-Quelltyp die Option Data stores (Datenspeicher) und anschließend Next (Weiter) aus.

5. Jetzt wollen wir den Crawler auf Ihre Daten verweisen. Wählen Sie auf der Seite Add a data store (Datenspeicher hinzufügen) den Datenspeicher „Amazon S3“ aus. In diesem Tutorial wird keine Verbindung verwendet. Lassen Sie daher das Feld Connection (Verbindung) leer, wenn Sie es sehen.

   Wählen Sie unter Crawl data in (Daten durchsuchen in) die Option Specified path in another account (Angegebener Pfad in einem anderen Konto) aus. Geben Sie dann bei Include path (Pfad hinzufügen) den Pfad ein, in dem der Crawler die Flugdaten finden kann. In diesem Fall ist es **`s3://crawler-public-us-east-1/flight/2016/csv`**. Nachdem Sie den Pfad eingegeben haben, ändert sich der Titel dieses Feldes zu Include path (Pfad hinzufügen). Wählen Sie Next (Weiter).

6. Sie können mehrere Datenspeicher mit einem einzigen Crawler durchsuchen. In diesem Tutorial verwenden wir jedoch nur einen einzelnen Datenspeicher. Wählen Sie daher No (Nein) aus und danach Next (Weiter).

7. Der Crawler benötigt Berechtigungen, um auf den Datenspeicher zuzugreifen und Objekte im AWS Glue Data Catalog. Um diese Berechtigungen zu konfigurieren, wählen Sie Create an IAM-Rolle. Der Name der IAM-Rolle beginnt mit `AWSGlueServiceRole-`. Im Feld geben Sie den letzten Teil des Rollennamens ein. Geben Sie **`CrawlerTutorial`** ein und wählen Sie Next (Weiter) aus.

> **Note**
>
> Um eine IAM-Rolle zu erstellen, muss Ihr AWS Benutzer die Berechtigungen `CreateRole`, `CreatePolicy` und `AttachRolePolicy` haben.

Der Assistent erstellt eine IAM-Rolle mit dem Namen`AWSGlueServiceRole-CrawlerTutorial`, hängt die AWS verwaltete Richtlinie an diese Rolle `AWSGlueServiceRole` an und fügt eine Inline-Richtlinie hinzu, die Lesezugriff auf den Amazon S3 S3-Standort ermöglicht. `s3://crawler-public-us-east-1/flight/2016/csv`

8. Erstellen Sie einen Zeitplan für den Crawler. Wählen Sie bei Frequency (Häufigkeit) die Option Run on demand (Bei Bedarf ausführen) aus und danach Next (Weiter).
9. Crawler erstellen in Ihrem Data Catalog Tabellen. Tabellen befinden sich in einer Datenbank im Data Catalog. Wählen Sie zuerst Add a databse (Datenbank hinzufügen), um eine Datenbank zu erstellen. Geben Sie im Popup-Fenster **`test-flights-db`** als Datenbankname ein und wählen Sie dann Create (Erstellen) aus.

   Geben Sie als Nächstes **`flights`** bei Prefix added to tables (Präfix für Tabellen) ein. Lassen Sie bei den restlichen Feldern die Standardwerte stehen und wählen Sie Next (Weiter) aus.
10. Überprüfen Sie die Eingaben im Assistenten Add crawler (Crawler hinzufügen). Wenn Sie Fehler bemerken, können Sie mit Back (Zurück) zu den vorherigen Seiten zurück und Änderungen vorzunehmen.

    Nachdem Sie alles überprüft haben, wählen Sie Finish (Abschließen) aus, um den Crawler zu erstellen.

Schritt 2: Den Crawler ausführen

Nachdem Sie einen Crawler erstellt haben, leitet Sie der Assistenten zur Ansicht des Crawlers weiter. Da Sie den Crawler mit einem On-Demand-Zeitplan erstellen, haben Sie die Möglichkeit, den Crawler auszuführen.

Den Crawler ausführen

1. Das Banner am oberen Rand dieser Seite zeigt Ihnen, dass der Crawler erstellt wurde. Es fragt, ob Sie ihn jetzt ausführen möchten. Wählen Sie Run it now? (Jetzt ausführen?) aus, um den Crawler auszuführen.

Das Banner ändert sich und zeigt jetzt die Meldungen „Attempting to run (Ausführung wird versucht)“ und „Running (Wird ausgeführt)“ für den Crawler an. Nachdem der Crawler gestartet wurde, verschwindet das Banner. Die Crawler-Anzeige ist nun aktualisiert und zeigt den Status „Starting (Wird gestartet)“. Nach einer Minute können Sie das Symbol zum Aktualisieren (Refresh) auswählen, um den aktuellen Status des Crawlers abzurufen, der in der Tabelle angezeigt wird.

2. Wenn der Crawler fertig ist, erscheint wieder ein Banner, das die vom Crawler vorgenommenen Änderungen beschreibt. Sie können den test-flights-dbLink wählen, um die Datenkatalogobjekte anzuzeigen.

Schritt 3: Ansehen AWS Glue Data Catalog objects

Der Crawler liest Daten am Quellspeicherort und erstellt Tabellen im Data Catalog. Eine Tabelle ist eine Metadatendefinition, die Ihre Daten sowie deren Schema repräsentiert. Die Tabellen im Data Catalog enthalten keine Daten. Stattdessen verwenden Sie diese Tabellen als Quelle oder Ziel bei Auftragsdefinitionen.

Die vom Crawler erstellten Data-Catalog-Objekte anzeigen

1. Wählen Sie im Navigationsbereich links unter Data catalog die Option Databases (Datenbanken) aus. Hier können Sie die Datenbank `flights-db` sehen, die vom Crawler erstellt wurde.
2. Wählen Sie im Navigationsbereich links unter Data catalog und unterhalb von Databases (Datenbanken) die Option Tables (Tabellen) aus. Hier können Sie die Tabelle `flightscsv` sehen, die vom Crawler erstellt wurde. Wenn Sie den Tabellennamen auswählen, können Sie die Tabelleneinstellungen, Parameter und Eigenschaften sehen. Wenn Sie in dieser Ansicht nach unten scrollen, sehen Sie das Schema, das Informationen zu den Spalten und Datentypen der Tabelle enthält.
3. Über View partitions (Partitionen anzeigen) auf der Tabellenansichtsseite können Sie die Partitionen sehen, die für die Daten erstellt wurden. In der ersten Spalte steht der Partitionsschlüssel.

Manuelles Definieren von Metadaten

Der AWS Glue Datenkatalog ist ein zentrales Repository, in dem Metadaten zu Ihren Datenquellen und Datensätzen gespeichert werden. Ein Crawler kann Metadaten für unterstützte Datenquellen

zwar automatisch crawlen und mit Daten füllen, es gibt jedoch bestimmte Szenarien, in denen Sie Metadaten möglicherweise manuell im Datenkatalog definieren müssen:

- Nicht unterstützte Datenformate — Wenn Sie Datenquellen haben, die vom Crawler nicht unterstützt werden, müssen Sie die Metadaten für diese Datenquellen manuell im Datenkatalog definieren.
- Anforderungen an benutzerdefinierte Metadaten — Der leitet Metadaten auf AWS-Glue-Crawler der Grundlage vordefinierter Regeln und Konventionen ab. Wenn Sie spezifische Metadatenanforderungen haben, die nicht von den AWS-Glue-Crawler abgeleiteten Metadaten abgedeckt werden, können Sie die Metadaten manuell definieren, um Ihren Anforderungen gerecht zu werden
- Datenverwaltung und Standardisierung — In einigen Fällen möchten Sie aus Gründen der Datenverwaltung, der Einhaltung von Vorschriften oder aus Sicherheitsgründen möglicherweise mehr Kontrolle über die Metadatendefinitionen haben. Durch die manuelle Definition von Metadaten können Sie sicherstellen, dass die Metadaten den Standards und Richtlinien Ihres Unternehmens entsprechen.
- Platzhalter für die future Datenaufnahme — Wenn Sie Datenquellen haben, die nicht sofort verfügbar oder zugänglich sind, können Sie leere Schematabellen als Platzhalter erstellen. Sobald die Datenquellen verfügbar sind, können Sie die Tabellen mit den tatsächlichen Daten füllen und dabei die vordefinierte Struktur beibehalten.

Um Metadaten manuell zu definieren, können Sie die AWS Glue Konsole, die Lake Formation Formation-Konsole, die AWS Glue API oder die AWS Command Line Interface (AWS CLI) verwenden. Sie können Datenbanken, Tabellen und Partitionen erstellen und Metadateneigenschaften wie Spaltennamen, Datentypen, Beschreibungen und andere Attribute angeben.

Erstellen von Datenbanken

Datenbanken werden verwendet, um Metadatentabellen in AWS Glue zu organisieren. Wenn Sie eine Tabelle in der definieren AWS Glue Data Catalog, fügen Sie sie einer Datenbank hinzu. Eine Tabelle kann sich jeweils nur in einer Datenbank befinden.

Ihre Datenbank kann Tabellen enthalten, die Daten aus vielen verschiedenen Datenspeichern definieren. Diese Daten können Objekte in Amazon Simple Storage Service (Amazon S3) und relationale Tabellen in Amazon Relational Database Service enthalten.

> **Note**
>
> Wenn Sie eine Datenbank aus dem löschen AWS Glue Datenkatalog, alle Tabellen in der Datenbank werden ebenfalls gelöscht.

Um die Liste der Datenbanken anzuzeigen, melden Sie sich bei der an AWS Management Console und öffnen Sie AWS Glue Konsole unter https://console.aws.amazon.com/glue/. Wählen Sie Databases (Datenbanken) aus und wählen Sie dann einen Datenbanknamen in der Liste aus, um die Details anzuzeigen.

Über die Registerkarte Datenbanken im AWS Glue In der Konsole können Sie Datenbanken hinzufügen, bearbeiten und löschen:

- Um eine neue Datenbank anzulegen, wählen Sie Add database (Datenbank hinzufügen) aus und geben Sie einen Namen und eine Beschreibung an. Aus Kompatibilitätsgründen mit anderen Metadatenspeichern, wie z. B. Apache Hive, wird der Name in Kleinbuchstaben gespeichert.

  > **Note**
  >
  > Wenn Sie über Amazon Athena auf die Datenbank zugreifen möchten, müssen Sie einen Namen angeben, der nur aus alphanumerischen Zeichen und Unterstrichen besteht. Weitere Informationen finden Sie unter Athena-Namen.

- Um die Beschreibung für eine Datenbank zu bearbeiten, aktivieren Sie das Kontrollkästchen neben dem Datenbanknamen und wählen Sie Edit (Bearbeiten) aus.
- Um eine Datenbank zu löschen, aktivieren Sie das Kontrollkästchen neben dem Datenbanknamen und wählen Sie Remove (Entfernen) aus.
- Um die Liste der in der Datenbank enthaltenen Tabellen anzuzeigen, wählen Sie den Datenbanknamen und die Datenbankeigenschaften zeigen alle Tabellen in der Datenbank an.

Um die Datenbank zu ändern, in die ein Crawler schreibt, müssen Sie die Crawler-Definition ändern. Weitere Informationen finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs .

Links zu Datenbankressourcen

> Das Tool AWS Glue Die Konsole wurde kürzlich aktualisiert. Die aktuelle Version der Konsole bietet keine Unterstützung für Datenbankressourcenlinks.

Außerdem kann der Data Catalog auch Ressourcen-Links zu Datenbanken enthalten. Eine Link zu einer Datenbankressource ist eine Verknüpfung zu einer lokalen oder freigegebenen Datenbank. Derzeit können Sie Ressourcenlinks nur in AWS Lake Formation erstellen. Nachdem Sie einen Ressourcenlink zu einer Datenbank erstellt haben, können Sie den Namen des Ressourcenlinks überall verwenden, wo Sie den Datenbanknamen verwenden möchten. Zusammen mit Datenbanken, die Ihnen gehören oder die mit Ihnen gemeinsam genutzt werden, werden Links zu Datenbankressourcen von zurückgegeben `glue:GetDatabases()` und als Einträge auf der Datenbankseite des AWS Glue console.

Außerdem kann der Data Catalog auch Ressourcenlinks zu Tabellen enthalten.

Weitere Informationen zu Ressourcenlinks finden Sie unter Creating Resource Links (Erstellen von Ressourcenlinks) im AWS Lake Formation -Entwicklerhandbuch.

Erstellen von Tabellen

Auch wenn das Ausführen eines Crawlers die empfohlene Methode ist, um die Daten in Ihren Datenspeichern zu inventarisieren, können Sie dem AWS Glue Data Catalog manuell Metadatentabellen hinzufügen. Dieser Ansatz ermöglicht es Ihnen, mehr Kontrolle über die Metadatendefinitionen zu haben und sie an Ihre spezifischen Anforderungen anzupassen.

Sie können dem Datenkatalog auf folgende Weise auch manuell Tabellen hinzufügen:

- Verwenden Sie die AWS Glue Konsole, um manuell eine Tabelle in der zu erstellen AWS Glue Data Catalog. Weitere Informationen finden Sie unter Erstellen von Tabellen mithilfe der Konsole.
- Verwenden Sie die `CreateTable`-Operation in der AWS Glue API, um eine Tabelle im AWS Glue Data Catalog anzulegen. Weitere Informationen finden Sie unter CreateTable Aktion (Python: create_table).
- Verwenden Sie AWS CloudFormation Vorlagen. Weitere Informationen finden Sie unter AWS CloudFormation für AWS Glue.

Wenn Sie eine Tabelle manuell mithilfe der Konsole oder einer API definieren, geben Sie das Tabellenschema und den Wert eines Klassifizierungsfeldes an, mit dem Typ und Format der Daten in der Datenquelle festgelegt werden. Wenn ein Crawler die Tabelle erstellt, werden das Datenformat und das Schema entweder durch einen integrierten Classifier oder einen angepassten Classifier bestimmt. Weitere Informationen zum Erstellen einer Tabelle mit dem AWS Glue Konsole finden Sie unterErstellen von Tabellen mithilfe der Konsole.

Themen

- Tabellenpartitionen
- Tabellenressourcen-Verknüpfungen
- Erstellen von Tabellen mithilfe der Konsole
- Partitionsindizes erstellen
- Aktualisieren von manuell erstellten Data-Catalog-Tabellen mit Crawlern
- Eigenschaften der Data-Catalog-Tabelle

Tabellenpartitionen

Importieren in &S3; AWS Glue Die Tabellendefinition eines Amazon Simple Storage Service (Amazon S3) -Ordners kann eine partitionierte Tabelle beschreiben. Um die Abfrageleistung zu verbessern, kann eine partitionierte Tabelle beispielsweise monatliche Daten in verschiedene Dateien unter Verwendung des Monatsnamens als Schlüssel aufteilen. In AWS Glue, Tabellendefinitionen beinhalten den Partitionierungsschlüssel einer Tabelle. Wann AWS Glue wertet die Daten in Amazon S3 S3-Ordnern aus, um eine Tabelle zu katalogisieren. Es bestimmt, ob eine einzelne Tabelle oder eine partitionierte Tabelle hinzugefügt wird.

Sie können Partitionsindizes für eine Tabelle erstellen, um eine Teilmenge der Partitionen abzurufen, anstatt alle Partitionen in der Tabelle zu laden. Weitere Informationen zum Arbeiten mit Indizes finden Sie unter Partitionsindizes erstellen .

Alle folgenden Bedingungen müssen zutreffen für AWS Glue um eine partitionierte Tabelle für einen Amazon S3 S3-Ordner zu erstellen:

- Die Schemas der Dateien sind ähnlich, bestimmt durch AWS Glue.
- Das Datenformat der Dateien ist identisch.
- Das Komprimierungsformat der Dateien ist identisch.

Sie könnten z. B. einen Amazon S3 Bucket namens `my-app-bucket` besitzen, in dem Sie sowohl iOS- als auch Android-App-Verkaufsdaten speichern. Die Daten werden nach Jahr, Monat und Tag partitioniert. Die Datendateien für iOS- und Android-Verkäufe haben das gleiche Schema, Datenformat und Komprimierungsformat. In der AWS Glue Data Catalog, der AWS Glue Der Crawler erstellt eine Tabellendefinition mit Partitionierungsschlüsseln für Jahr, Monat und Tag.

Die folgende Amazon-S3-Auflistung von `my-app-bucket` zeigt einige der Partitionen. Das =-Symbol dient zur Zuweisung von Partitionsschlüsselwerten.

```
my-app-bucket/Sales/year=2010/month=feb/day=1/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=1/Android.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/Android.csv
...
my-app-bucket/Sales/year=2017/month=feb/day=4/iOS.csv
my-app-bucket/Sales/year=2017/month=feb/day=4/Android.csv
```

Tabellenressourcen-Verknüpfungen

> Das Tool AWS Glue Die Konsole wurde kürzlich aktualisiert. Die aktuelle Version der Konsole bietet keine Unterstützung für Tabellenressourcenlinks.

Außerdem kann der Data Catalog auch Ressourcenlinks zu Tabellen enthalten. Ein Link zu einer Tabellenressource ist eine Verknüpfung zu einer lokalen oder freigegebenen Datenbank. Derzeit können Sie Ressourcenlinks nur in AWS Lake Formation erstellen. Nachdem Sie einen Ressourcenlink zu einer Tabelle erstellt haben, können Sie den Namen des Ressourcenlinks überall verwenden, wo Sie den Tabellennamen verwenden möchten. Zusammen mit Tabellen, die Sie besitzen oder die für Sie gemeinsam genutzt werden, werden Tabellenressourcen-Links von zurückgegeben `glue:GetTables()` und als Einträge auf der Tabellenseite des AWS Glue console.

Außerdem kann der Data Catalog auch Ressourcen-Links zu Datenbanken enthalten.

Weitere Informationen zu Ressourcenlinks finden Sie unter Creating Resource Links (Erstellen von Ressourcenlinks) im AWS Lake Formation -Entwicklerhandbuch.

Erstellen von Tabellen mithilfe der Konsole

Eine Tabelle in der AWS Glue Data Catalog ist die Metadatendefinition, die die Daten in einem Datenspeicher darstellt. Sie erstellen Tabellen, wenn Sie einen Crawler ausführen, oder Sie können eine Tabelle manuell in der AWS Glue -Konsole erstellen. Die Tabellenliste in der AWS Glue Die Konsole zeigt Werte der Metadaten Ihrer Tabelle an. Sie verwenden Tabellendefinitionen zum Angeben von Quellen und Zielen, wenn Sie ETL (Extrahieren, Transformieren und Laden)-Aufträge erstellen.

> **Note**
>
> Aufgrund der jüngsten Änderungen an der AWS Managementkonsole müssen Sie möglicherweise Ihre vorhandenen IAM-Rollen ändern, um die `SearchTables`entsprechende Berechtigung zu erhalten. Für die Erstellung neuer Rollen wurde die `SearchTables`-API-Berechtigung bereits standardmäßig hinzugefügt.

Melden Sie sich zunächst bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/. Wählen Sie die Registerkarte Tabellen aus und verwenden Sie die Add tables (Tabellen hinzufügen)-Schaltfläche, um Tabellen entweder mit einem Crawler oder durch die manuelle Eingabe von Attributen zu erstellen.

Hinzufügen von Tabellen in der Konsole

Um einen Crawler zum Hinzufügen von Tabellen zu verwenden, wählen Sie Add tables (Tabellen hinzufügen), Add tables using a crawler (Tabellen mit einem Crawler hinzufügen) aus. Dann folgen Sie den Anweisungen im Add crawler (Crawler hinzufügen)-Assistenten. Wenn der Crawler ausgeführt wird, werden Tabellen dem AWS Glue Data Catalog hinzugefügt. Weitere Informationen finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs .

Wenn Sie die Attribute kennen, die zum Erstellen einer Amazon Simple Storage Service (Amazon S3)-Tabellendefinition in Ihrem Data Catalog erforderlich sind, können Sie sie mit dem Tabellenassistenten erstellen. Wählen Sie Add tables (Tabellen hinzufügen), Add table manually (Tabelle manuell hinzufügen) aus und befolgen Sie die Anweisungen im Add table (Tabelle hinzufügen)-Assistenten.

Wenn eine Tabelle über die Konsole manuell hinzufügt wird, sollten Sie Folgendes beachten:

- Wenn Sie über Amazon Athena auf die Tabelle zugreifen möchten, müssen Sie einen Namen angeben, der nur aus alphanumerischen Zeichen und Unterstrichen besteht. Weitere Informationen finden Sie unter Athena-Namen.
- Der Speicherort der Quelldaten muss ein Amazon-S3-Pfad sein.
- Das Datenformat der Daten muss mit einem der aufgeführten Formate im Assistenten übereinstimmen. Die entsprechende Klassifizierung und andere Tabelleneigenschaften werden automatisch auf der Grundlage des ausgewählten Formats ausgefüllt. SerDe Sie können Tabellen mit den folgenden Formaten definieren:

  Avro

  Apache Avro-JSON-Binärformat.

  CSV

  Werte mit Zeichentrennung. Außerdem geben Sie das Trennzeichen entweder als Komma, Pipe, Semikolon, Tabstopp oder Strg-A an.

  JSON

  JavaScript Objektnotation.

  XML

  Extensible Markup Language-Format. Geben Sie das XML-Tag an, das eine Zeile in den Daten definiert. Spalten werden in Zeilen-Tags definiert.

  Parquet

  Spaltenweise Speicherung von Apache Parquet.

  ORC

  Optimiertes ORC-Dateiformat (Row Columnar). Ein Format zur effizienten Speicherung von Hive-Daten.
- Sie können einen Partitionsschlüssel für die Tabelle definieren.
- Derzeit können partitionierte Tabellen, die Sie mit der Konsole erstellen, nicht in ETL-Aufträgen verwendet werden.

Tabellenattribute

Es folgen einige wichtige Attribute Ihrer Tabelle:

Name

Der Name wird festgelegt, wenn die Tabelle erstellt wird, und kann nicht geändert werden. In vielen Fällen beziehen Sie sich auf einen Tabellennamen AWS Glue Operationen.

Datenbank

Das Container-Objekt, in dem die Tabelle gespeichert ist. Dieses Objekt enthält eine Organisation Ihrer Tabellen, die innerhalb des Datenspeichers existiert AWS Glue Data Catalog und sich von einer Organisation in Ihrem Datenspeicher unterscheiden kann. Wenn Sie eine Datenbank löschen, werden alle Tabellen in der Datenbank ebenfalls von dem Data Catalog gelöscht.

Beschreibung

Die Beschreibung der Tabelle. Sie können eine Beschreibung zum besseren Verständnis der Inhalte der Tabelle schreiben.

Tabellenformat

Geben Sie an, eine AWS Glue Standardtabelle oder eine Tabelle im Apache Iceberg-Format zu erstellen.

Der Datenkatalog bietet die folgenden Optionen zur Tabellenoptimierung, um den Tabellenspeicher zu verwalten und die Abfrageleistung für Iceberg-Tabellen zu verbessern.

- Komprimierung — Datendateien werden zusammengeführt und neu geschrieben, veraltete Daten werden entfernt und fragmentierte Daten werden in größeren, effizienteren Dateien konsolidiert.
- Aufbewahrung von Snapshots — Snapshots sind Versionen einer Iceberg-Tabelle mit Zeitstempel. Mit Konfigurationen zur Aufbewahrung von Snapshots können Kunden festlegen, wie lange und wie viele Snapshots aufbewahrt werden sollen. Die Konfiguration eines Optimizers für die Aufbewahrung von Snapshots kann helfen, den Speicheraufwand zu minimieren, indem ältere, unnötige Snapshots und die zugehörigen zugrunde liegenden Dateien entfernt werden.
- Löschen verwaister Dateien — Verwaiste Dateien sind Dateien, auf die in den Metadaten der Iceberg-Tabelle nicht mehr verwiesen wird. Diese Dateien können sich im Laufe der Zeit ansammeln, insbesondere nach Vorgängen wie dem Löschen von Tabellen oder fehlgeschlagenen ETL-Jobs. Wenn Sie das Löschen verwaister Dateien aktivieren AWS Glue , können Sie diese unnötigen Dateien regelmäßig identifizieren und entfernen, wodurch Speicherplatz frei wird.

Weitere Informationen finden Sie unter Optimieren von Iceberg-Tabellen.

Konfiguration der Optimierung

Sie können entweder die Standardeinstellungen verwenden oder die Einstellungen für die Aktivierung der Tabellenoptimierer anpassen.

IAM-Rolle

Um die Tabellenoptimierer auszuführen, übernimmt der Dienst in Ihrem Namen eine IAM-Rolle. Sie können über das Dropdown-Menü eine IAM-Rolle auswählen. Die Rolle sollte die erforderlichen Berechtigungen für die Verdichtung haben.

Weitere Informationen zu den erforderlichen Berechtigungen für die IAM-Rolle finden Sie unter Voraussetzungen für die Tabellenoptimierung .

Ort

Der Zeiger auf den Speicherort der Daten in einem Datenspeicher, den diese Tabellendefinition repräsentiert.

Klassifizierung

Ein Kategorisierungswert, der bei der Erstellung der Tabelle bereitgestellt wurde. In der Regel wird dieser geschrieben, wenn ein Crawler ausgeführt wird, und gibt das Format der Quelldaten an.

Letzte Aktualisierung

Die Uhrzeit und das Datum (UTC), zu denen diese Tabelle im Data Catalog aktualisiert wurde.

Datum hinzugefügt

Die Uhrzeit und das Datum (UTC), zu denen diese Tabelle dem Data Catalog hinzugefügt wurde.

Als veraltet gekennzeichnet

Wenn AWS Glue stellt fest, dass eine Tabelle im Datenkatalog nicht mehr in ihrem ursprünglichen Datenspeicher vorhanden ist, und markiert die Tabelle im Datenkatalog als veraltet. Wenn Sie einen Auftrag ausführen, der auf eine veraltete Tabelle verweist, kann der Auftrag fehlschlagen. Bearbeiten Sie Aufträge, die auf veraltete Tabellen verweisen, um sie als Quellen und Ziele zu entfernen. Wir empfehlen, dass Sie veraltete Tabellen löschen, wenn sie nicht mehr benötigt werden.

Verbindung

Wenn AWS Glue erfordert eine Verbindung zu Ihrem Datenspeicher, der Name der Verbindung ist der Tabelle zugeordnet.

Tabellendetails anzeigen und verwalten

Um die Details einer vorhandenen Tabelle anzuzeigen, wählen Sie den Tabellennamen in der Liste und dann Action, View details (Aktion, Details anzeigen) aus.

Die Tabellendetails umfassen Eigenschaften der Tabelle und deren Schema. Diese Ansicht zeigt das Schema der Tabelle an, einschließlich Spaltennamen in der Reihenfolge, die für die Tabelle, Datentypen und Schlüsselspalten für Partitionen definiert wurde. Wenn eine Spalte ein komplexer Typ ist, können Sie View properties (Eigenschaften anzeigen) auswählen, um Details der Struktur dieses Felds anzuzeigen, wie im folgenden Beispiel dargestellt:

```
{
"StorageDescriptor":
    {
      "cols": {
         "FieldSchema": [
           {
             "name": "primary-1",
             "type": "CHAR",
             "comment": ""
           },
           {
             "name": "second ",
             "type": "STRING",
             "comment": ""
           }
         ]
      },
      "location": "s3://aws-logs-111122223333-us-east-1",
      "inputFormat": "",
      "outputFormat": "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
      "compressed": "false",
      "numBuckets": "0",
      "SerDeInfo": {
           "name": "",
           "serializationLib": "org.apache.hadoop.hive.serde2.OpenCSVSerde",
           "parameters": {
               "separatorChar": "|"
            }
      },
      "bucketCols": [],
      "sortCols": [],
      "parameters": {},
```

```
      "SkewedInfo": {},
      "storedAsSubDirectories": "false"
    },
    "parameters": {
       "classification": "csv"
    }
}
```

Weitere Informationen zu den Eigenschaften einer Tabelle, wie z. B. `StorageDescriptor`, finden Sie unter StorageDescriptor Struktur.

Wenn Sie das Schema einer Tabelle ändern möchten, wählen Sie Edit schema (Schema bearbeiten) aus, um Spalten hinzuzufügen oder zu löschen und Spaltennamen und Datentypen zu ändern.

Um verschiedene Versionen einer Tabelle, einschließlich ihres Schemas, zu vergleichen, wählen Sie Versionen vergleichen aus, um einen side-by-side Vergleich zweier Versionen des Schemas für eine Tabelle anzuzeigen. Weitere Informationen finden Sie unter Vergleich der Versionen von Tabellenschemas .

Zum Anzeigen der Dateien, aus denen eine Amazon-S3-Partition besteht, wählen Sie View Partition (Partition anzeigen) aus. Bei Amazon-S3-Tabellen zeigt die Schlüssel-Spalte die Partitionsschlüssel an, die verwendet werden, um die Tabelle im Quelldatenspeicher zu partitionieren. Die Partitionierung ist eine Möglichkeit zum Aufteilen einer Tabelle in verknüpfte Teile basierend auf den Werten einer Schlüsselspalte wie beispielsweise Datum, Ort oder Abteilung. Für weitere Informationen zu Partitionen, suchen Sie im Internet nach Informationen über "Hive-Partitionierung".

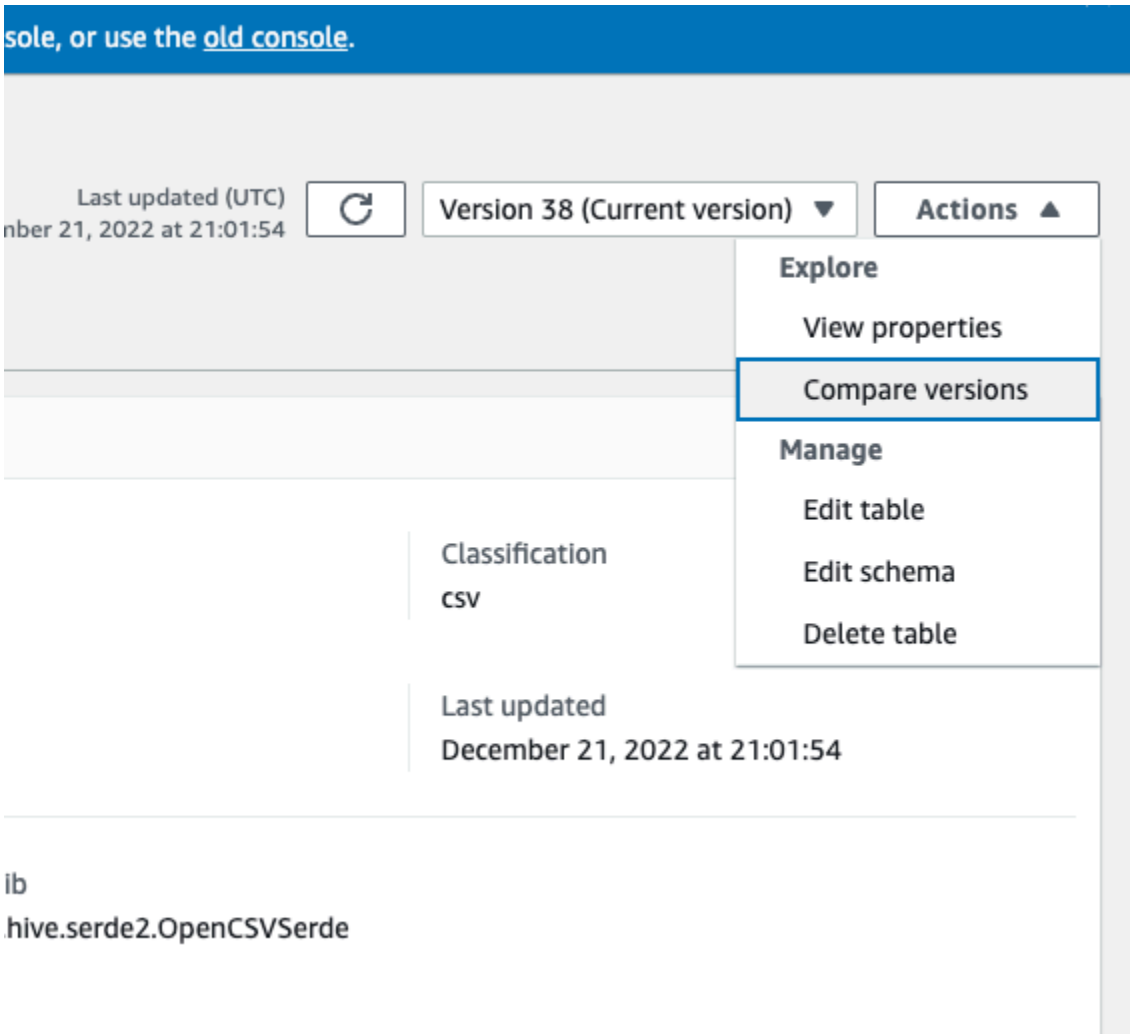
> **Note**
>
> step-by-stepAnleitungen zum Anzeigen der Details einer Tabelle finden Sie in der Konsole im Tutorial „Tabelle erkunden“.

Vergleich der Versionen von Tabellenschemas

Wenn Sie zwei Versionen von Tabellenschemas vergleichen, können Sie Änderungen an verschachtelten Zeilen vergleichen, indem Sie verschachtelte Zeilen erweitern und reduzieren, Schemas zweier Versionen side-by-side vergleichen und Tabelleneigenschaften anzeigen. side-by-side

Wie Sie Versionen vergleichen

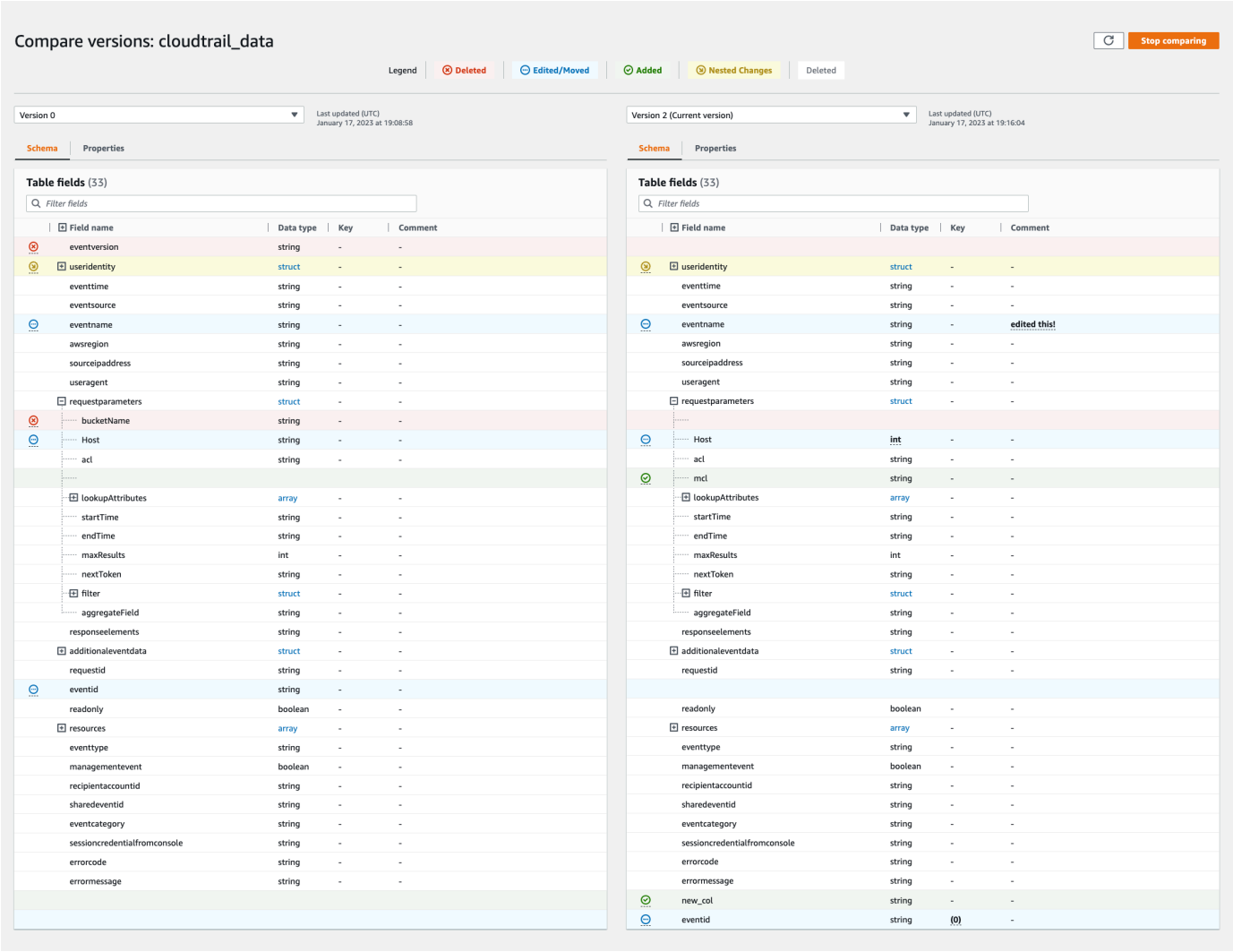
1. Aus dem AWS Glue Wählen Sie in der Konsole Tabellen, dann Aktionen und dann Versionen vergleichen aus.



2. Wählen Sie eine Version, die Sie vergleichen möchten, indem Sie das Dropdown-Menü wählen. Beim Vergleich von Schemas ist die Registerkarte Schema orange hervorgehoben.
3. Wenn Sie Tabellen zwischen zwei Versionen vergleichen, werden Ihnen die Tabellenschemas auf der linken und rechten Seite des Bildschirms angezeigt. Auf diese Weise können Sie Änderungen visuell feststellen, indem Sie die Felder Spaltenname, Datentyp, Schlüssel und Kommentar vergleichen side-by-side. Wenn es eine Änderung gibt, zeigt ein farbiges Symbol die Art der vorgenommenen Änderung an.

   - Gelöscht – ein rotes Symbol zeigt an, wo die Spalte aus einer früheren Version des Tabellenschemas entfernt wurde.
   - Bearbeitet oder Verschoben – Ein blaues Symbol zeigt an, wo die Spalte in einer neueren Version des Tabellenschemas geändert oder verschoben wurde.

- Gelöscht – ein rotes Symbol zeigt an, wo die Spalte aus einer früheren Version des Tabellenschemas entfernt wurde.
- Verschachtelte Änderungen – Ein gelbes Symbol zeigt an, wo die verschachtelte Spalte Änderungen enthält. Wählen Sie die Spalte aus, die erweitert werden soll, und sehen Sie sich die Spalten an, die entweder gelöscht, bearbeitet, verschoben oder hinzugefügt wurden.



4. Verwenden Sie die Suchleiste für Filterfelder, um Felder anzuzeigen, die auf den Zeichen basieren, die Sie hier eingeben. Wenn Sie in einer der Tabellenversionen einen Spaltennamen eingeben, werden die gefilterten Felder in beiden Tabellenversionen angezeigt, um Ihnen zu zeigen, wo die Änderungen vorgenommen wurden.
5. Um Eigenschaften zu vergleichen, wählen Sie die Registerkarte der Eigenschaften.
6. Um den Versionsvergleich zu beenden, wählen Sie Vergleich beenden, um zur Liste der Tabellen zurückzukehren.

Partitionsindizes erstellen

Im Laufe der Zeit werden Hunderttausende von Partitionen zu einer Tabelle hinzugefügt. Die GetPartitions API wird verwendet, um die Partitionen in der Tabelle abzurufen. Die API gibt Partitionen zurück, die dem in der Anfrage angegebenen Ausdruck entsprechen.

Nehmen wir als Beispiel eine sales_data-Tabelle, die nach den Schlüsseln Country, Category, Year, Month und CreationDate partitioniert ist. Wenn Sie Verkaufsdaten für alle Artikel erhalten möchten, die im Jahr 2020 nach dem 15.08.2020 für die Kategorie Bücher verkauft wurden, müssen Sie eine `GetPartitions` Anfrage mit dem Ausdruck „Kategorie = 'Bücher' und CreationDate > '2020-08-15'“ an den Datenkatalog stellen.

Wenn in der Tabelle keine Partitionsindizes vorhanden sind, AWS Glue lädt alle Partitionen der Tabelle und filtert dann die geladenen Partitionen anhand des vom Benutzer in der `GetPartitions` Anforderung angegebenen Abfrageausdrucks. Die Ausführung der Abfrage nimmt mehr Zeit in Anspruch, wenn die Anzahl der Partitionen in einer Tabelle ohne Indizes zunimmt. Mit einem Index versucht die `GetPartitions`-Abfrage eine Teilmenge der Partitionen abzurufen, statt alle Partitionen in der Tabelle zu laden.

Themen

- Informationen zu Partitionsindizes
- Erstellen einer Tabelle mit Partitionsindizes
- Hinzufügen eines Partitionsindexes zu einer vorhandenen Tabelle
- Beschreiben von Partitionsindizes in einer Tabelle
- Einschränkungen bei der Verwendung von Partitionsindizes
- Verwenden von Indizes für einen optimierten Anruf GetPartitions
- Integration mit Engines

Informationen zu Partitionsindizes

Wenn Sie einen Partitionsindex erstellen, geben Sie eine Liste der Partitionsschlüssel an, die bereits in einer bestimmten Tabelle vorhanden sind. Partitionsindex ist eine untergeordnete Liste der Partitionsschlüssel, die in der Tabelle definiert sind. Ein Partitionsindex kann auf jeder Permutation von Partitionsschlüsseln erstellt werden, die in der Tabelle definiert sind. Für die obige sales_data-Tabelle sind die möglichen Indizes (Land, Kategorie, creationDate), (Land, Kategorie, Jahr), (Land, Kategorie), (Land), (Kategorie, Land, Jahr, Monat) usw. möglich.

Der Data Catalog verkettet die Partitionswerte in der Reihenfolge, die zum Zeitpunkt der Indexerstellung angegeben wurde. Der Index wird konsistent erstellt, wenn Partitionen zur Tabelle hinzugefügt werden. Indizes können für die Spaltentypen String (string, char und varchar), Numeric (int, bigint, long, tinyint und smallint) und Date (yyyy-MM-DD) erstellt werden.

Unterstützte Datentypen

- Datum — Ein Datum im ISO-Format, z. `YYYY-MM-DD` B. Zum Beispiel Datum`2020-08-15`. Das Format verwendet Bindestriche (-), um Jahr, Monat und Tag zu trennen. Der zulässige Bereich für Datumsangaben für die Indizierung reicht von bis. `0000-01-01` `9999-12-31`
- Zeichenfolge — Ein Zeichenfolgenliteral, das in einfache oder doppelte Anführungszeichen eingeschlossen ist.
- Char — Zeichendaten mit fester Länge und einer angegebenen Länge zwischen 1 und 255, z. B. char (10).
- Varchar — Zeichendaten mit variabler Länge und einer angegebenen Länge zwischen 1 und 65535, z. B. varchar (10).
- Numerisch — int, bigint, long, tinyint und smallint

Indizes für die Datentypen Numeric, String und Date unterstützen die Operatoren `=`, `>`, `>=`, `<`, `<=` und zwischen. Die Indexierungslösung unterstützt derzeit nur den logischen Operator `AND`. Subausdrücke mit den Operatoren „LIKE“, „IN“, „OR“ und „NOT“ werden im Ausdruck zum Filtern mit einem Index ignoriert. Die Filterung für den ignorierten Subausdruck erfolgt auf den Partitionen, die nach dem Anwenden der Indexfilterung abgerufen werden.

Für jede Partition, die einer Tabelle hinzugefügt wird, wird ein entsprechendes Indexelement erstellt. Für eine Tabelle mit 'n'-Partitionen führt 1 Partitionsindex zu 'n' Partitionsindex-Elementen. Der Partitionsindex „m“ in derselben Tabelle führt zu „m*n“ Partitionsindex-Elementen. Für jedes Element im Partitionsindex wird der aktuelle Wert berechnet AWS Glue Preispolitik für die Speicherung von Datenkatalogen. Einzelheiten zur Preisgestaltung von Speicherobjekten finden Sie unter AWS Glue Preisgestaltung.

Erstellen einer Tabelle mit Partitionsindizes

Sie können einen Partitionsindex während der Tabellenerstellung erstellen. Die `CreateTable`-Anforderung nimmt eine Liste von `PartitionIndex`-Objekten als Eingabe. Für eine bestimmte Tabelle können maximal 3 Partitionsindizes erstellt werden. Jeder Partitionsindex erfordert einen

Namen und eine Liste von `partitionKeys`, die für die Tabelle definiert ist. Erstellte Indizes für eine Tabelle können mit der `GetPartitionIndexes-API` abgerufen werden

Hinzufügen eines Partitionsindexes zu einer vorhandenen Tabelle

Um einen Partitionsindex einer vorhandenen Tabelle hinzuzufügen, verwenden Sie die `CreatePartitionIndex`-Operation. Sie können einen `PartitionIndex` pro `CreatePartitionIndex`-Operation erstellen. Das Hinzufügen eines Index wirkt sich nicht auf die Verfügbarkeit einer Tabelle aus, da die Tabelle während der Erstellung von Indizes weiterhin verfügbar ist.

Der Indexstatus für eine hinzugefügte Partition wird auf CREATING gesetzt und die Erstellung der Indexdaten wird gestartet. Wenn der Prozess zum Erstellen der Indizes erfolgreich ist, wird der indexStatus auf ACTIVE aktualisiert, und für einen nicht erfolgreichen Prozess wird der Indexstatus auf FAILED aktualisiert. Die Indexerstellung kann aus mehreren Gründen fehlschlagen, und Sie können die `GetPartitionIndexes`-Operation verwenden, um die Fehlerdetails abzurufen. Mögliche Fehler:

- ENCRYPTED_PARTITION_ERROR – Die Indexerstellung für eine Tabelle mit verschlüsselten Partitionen wird nicht unterstützt.
- INVALID_PARTITION_TYPE_DATA_ERROR – Wird beobachtet, wenn der `partitionKey`-Wert kein gültiger Wert für den entsprechenden `partitionKey`-Datentyp ist. Beispiel: ein `partitionKey` mit dem Datentyp 'int' hat einen Wert 'foo'.
- MISSING_PARTITION_VALUE_ERROR – Tritt auf, wenn der `partitionValue` für einen `indexedKey` nicht vorhanden ist. Dies kann passieren, wenn eine Tabelle nicht konsistent partitioniert wird.
- UNSUPPORTED_PARTITION_CHARACTER_ERROR – Wird beobachtet, wenn der Wert für einen indizierten Partitionsschlüssel die Zeichen \u0000, \u0001 oder \u0002 enthält
- INTERNAL_ERROR – Beim Erstellen von Indizes ist ein interner Fehler aufgetreten.

Beschreiben von Partitionsindizes in einer Tabelle

Verwenden Sie die `GetPartitionIndexes`-Operation, um die für eine Tabelle erstellten Partitionsindizes abzurufen. Die Antwort zeigt alle Indizes der Tabelle, zusammen mit deren jeweiligem aktuellen Status (dem `IndexStatus`).

Der `IndexStatus` für einen Partitionsindex ist einer der Folgenden:

- `CREATING` – Der Index wird derzeit erstellt und ist noch nicht verfügbar.
- `ACTIVE` – Der Index ist jetzt verfügbar. Anforderungen können den Index verwenden, um eine optimierte Abfrage auszuführen.
- `DELETING` – Der Index wird derzeit gelöscht und kann nicht länger verwendet werden. Ein Index im aktiven Zustand kann mit der `DeletePartitionIndex`-Anforderung gelöscht werden, die den Status von AKTIV zu LÖSCHEN ändert.
- `FAILED` – Die Indexerstellung für eine vorhandene Tabelle ist fehlgeschlagen. In jeder Tabelle werden die letzten 10 fehlgeschlagenen Indizes gespeichert.

Die möglichen Zustandsübergänge für Indizes, die für eine vorhandene Tabelle erstellt wurden, sind:

- ERSTELLEN → AKTIV → LÖSCHEN
- ERSTELLEN → FEHLGESCHLAGEN

Einschränkungen bei der Verwendung von Partitionsindizes

Nachdem Sie einen Partitionsindex erstellt haben, beachten Sie die folgenden Änderungen an der Tabellen- und Partitionsfunktionalität:

Erstellung einer neuen Partition (nach dem Hinzufügen des Index)

Nachdem ein Partitionsindex für eine Tabelle erstellt wurde, werden alle neuen Partitionen, die der Tabelle hinzugefügt wurden, für die Datentypüberprüfungen für indizierte Schlüssel validiert. Der Partitionswert der indizierten Schlüssel wird für das Datentypformat validiert. Wenn die Datentypprüfung fehlschlägt, schlägt der Vorgang zum Erstellen der Partition fehl. Das Erstellen einer neuen Partition mit dem Wert JAHR als „foo“ schlägt für die sales_data-Tabelle fehl, wenn ein Index für die Schlüssel (Kategorie, Jahr) erstellt wird, wobei die Kategorie vom Typ `string` und Jahr vom Typ `int` ist.

Nachdem Indizes aktiviert sind, schlägt das Hinzufügen von Partitionen mit indizierten Schlüsselwerten mit den Zeichen U+0000, U+00001 und U+0002 fehl.

Tabellen-Aktualisierungen

Sobald ein Partitionsindex für eine Tabelle erstellt wurde, können Sie die Partitionsschlüsselnamen für vorhandene Partitionsschlüssel nicht ändern. Das gleiche gilt für den Typ oder die Reihenfolge der Schlüssel, die mit dem Index registriert sind.

Verwenden von Indizes für einen optimierten Anruf GetPartitions

Wenn Sie `GetPartitions` für eine Tabelle mit einem Index verwenden, können Sie einen Ausdruck einschließen, und wenn möglich, verwendet der Data Catalog einen Index. Der erste Schlüssel des Indexes sollte im Ausdruck für die Indizes übergeben werden, die beim Filtern verwendet werden sollen. Die Indexoptimierung in der Filterung wird als Best Effort angewendet. Der Data Catalog versucht, die Indexoptimierung so weit wie möglich zu verwenden, aber im Falle eines fehlenden Index oder nicht unterstützten Operators fällt er auf die vorhandene Implementierung des Ladens aller Partitionen zurück.

Fügen wir für die sales_data-Tabelle oben den Index [Land, Kategorie, Jahr] hinzu. Wenn „Country“ im Ausdruck nicht übergeben wird, kann der registrierte Index keine Partitionen mithilfe von Indizes filtern. Sie können bis zu 3 Indizes hinzufügen, um verschiedene Abfragemuster zu unterstützen.

Nehmen wir einige Beispielausdrücke und sehen, wie Indizes darauf funktionieren:

Ausdrücke	Wie wird der Index verwendet
Land = „USA“	Index wird verwendet, um Partitionen zu filtern.
Land = 'USA' und Kategorie = 'Schuhe'	Index wird verwendet, um Partitionen zu filtern.
Kategorie = 'Schuhe'	Indizes werden nicht verwendet, da „Land“ nicht im Ausdruck angegeben wird. Alle Partitionen werden geladen, um eine Antwort zurückzugeben.
Land = 'USA' und Kategorie = 'Schuhe' und Jahr > '2018'	Index wird verwendet, um Partitionen zu filtern.
Land = 'USA' und Kategorie = 'Schuhe' und Jahr > '2018' und Monat = 2	Index wird verwendet, um alle Partitionen abzurufen; Land = „USA“ und Kategorie = „Schuhe“ und Jahr > 2018. Anschließend wird nach dem Ausdruck 'Monat' gefiltert.
Land = 'USA' UND Kategorie = 'Schuhe' ODER Jahr > '2018'	Indizes werden nicht verwendet, da ein `OR`-Operator im Ausdruck vorhanden ist.
Land = 'USA' UND Kategorie = 'Schuhe' UND (Jahr > 2017 ODER Jahr = '2018')	Index wird verwendet, um alle Partitionen mit Land = 'USA' und Kategorie = 'Schuhe'

Ausdrücke	Wie wird der Index verwendet
	abzurufen. Anschließend wird nach dem Ausdruck 'Jahr' gefiltert.
Land ist ('USA', 'GB') UND Kategorie = 'Schuhe'	Indizes werden nicht zum Filtern verwendet, da der `IN`-Operator derzeit nicht unterstützt wird.
Land = 'USA' UND Kategorie in ('Schuhe', 'Bücher')	Index wird verwendet, um alle Partitionen mit Land = "USA" abzurufen, und dann wird eine Filterung nach dem Kategorieausdruck durchgeführt.
Land = 'USA' UND Kategorie in ('Schuhe', 'Bücher') UND (creationDate > '2023-9-01'	Der Index wird verwendet, um alle Partition en mit country = „US“ und mit CreationDate > '2023-9-01' abzurufen. Anschließend wird nach dem Categorie-Ausdruck gefiltert.

Integration mit Engines

Redshift Spectrum, Amazon EMR und AWS Glue ETL Spark DataFrames ist in der Lage, Indizes zum Abrufen von Partitionen zu verwenden, nachdem sich Indizes im Status AKTIV befinden in AWS Glue. Athena und AWS Glue Bei dynamischen ETL-Frames müssen Sie zusätzliche Schritte ausführen, um Indizes zur Verbesserung von Abfragen zu verwenden.

Aktivieren Sie die Partitionsfilterung

Um die Partitionsfilterung in Athena zu aktivieren, müssen Sie die Tabelleneigenschaften wie folgt aktualisieren:

1. Wählen Sie in der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie eine -Tabelle aus.
3. Wählen Sie unter Aktionen die Option Tabelle bearbeiten aus.
4. Fügen Sie unter Tabelleneigenschaften Folgendes hinzu:
   - Schlüssel — `partition_filtering.enabled`
   - Wert — `true`
5. Wählen Sie Anwenden aus.

Alternativ können Sie diesen Parameter festlegen, indem Sie eine ALTER TABLE SET PROPERTIES-Abfrage in Athena ausführen.

```
ALTER TABLE partition_index.table_with_index
SET TBLPROPERTIES ('partition_filtering.enabled' = 'true')
```

Aktualisieren von manuell erstellten Data-Catalog-Tabellen mit Crawlern

Möglicherweise möchten Sie AWS Glue Data Catalog Tabellen manuell erstellen und sie dann auf dem neuesten Stand halten AWS Glue Crawler. Crawler, die nach einem Zeitplan ausgeführt werden, können neue Partitionen hinzufügen und die Tabellen mit allen Schemaänderungen aktualisieren. Dies gilt auch für Tabellen, die aus aus einem Apache Hive-Metastore migriert wurden.

Hierzu geben Sie beim Definieren eines Crawlers statt eines oder mehrerer Datenspeicher als Quelle eines Crawls eine oder mehrere vorhandene Data-Catalog-Tabellen an. Der Crawler durchsucht dann die durch die Katalogtabellen angegebenen Datenspeicher. In diesem Fall werden keine neuen Tabellen erstellt. Stattdessen werden Ihre manuell erstellten Tabellen aktualisiert.

Es folgen weitere mögliche Gründe dafür, Katalogtabellen manuell zu erstellen und Katalogtabellen als Crawler-Quelle anzugeben.

- Sie möchten den Katalog-Tabellennamen wählen und dies nicht dem Benennungsalgorithmus der Katalogtabelle überlassen.
- Sie möchten verhindern, dass neue Tabellen erstellt werden, falls Dateien mit einem die Paritionserkennung störenden Format versehentlich im Pfad der Datenquelle gespeichert werden.

Weitere Informationen finden Sie unter Schritt 2: Auswahl von Datenquellen und Classifier.

Eigenschaften der Data-Catalog-Tabelle

Tabelleneigenschaften oder Parameter, wie sie in der AWS CLI genannt werden, sind nicht validierte Schlüssel- und Wertezeichenfolgen. Sie können Ihre eigenen Eigenschaften für die Tabelle festlegen, um die Verwendung des Data Catalog außerhalb von AWS Glue zu unterstützen. Andere Dienste, die den Datenkatalog verwenden, können dies ebenfalls tun. AWS Glue legt einige Tabelleneigenschaften fest, wenn Jobs oder Crawler ausgeführt werden. Sofern nicht anders beschrieben, sind diese Eigenschaften für den internen Gebrauch bestimmt. Wir unterstützen nicht, dass sie in ihrer aktuellen Form fortbestehen, und wir unterstützen auch nicht das Produktverhalten, wenn diese Eigenschaften manuell geändert werden.

Weitere Hinweise zu Tabelleneigenschaften, die von AWS Glue Crawlern festgelegt wurden, finden Sie unter. the section called “Parameter, die vom Crawler in Data-Catalog-Tabellen festgelegt wurden”

Integration mit anderen AWS Diensten

Sie können zwar AWS-Glue-Crawler s verwenden, um sie zu füllen AWS Glue Data Catalog, aber es gibt mehrere AWS Dienste, die sich automatisch in den Katalog integrieren und ihn für Sie füllen können. In den folgenden Abschnitten finden Sie weitere Informationen zu den spezifischen Anwendungsfällen, die von AWS Diensten unterstützt werden, die den Datenkatalog auffüllen können.

Themen

- AWS Lake Formation
- Amazon Athena

AWS Lake Formation

AWS Lake Formation ist ein Dienst, der es einfacher macht, einen sicheren Data Lake einzurichten. AWS Lake Formation ist darauf aufgebaut AWS Glue, und Lake Formation und AWS Glue ich teilen dasselbe AWS Glue Data Catalog. Sie können Ihren Amazon S3 S3-Datenstandort bei Lake Formation registrieren und die Lake Formation Formation-Konsole verwenden, um Datenbanken und Tabellen im AWS Glue Datenkatalog zu erstellen, Datenzugriffsrichtlinien zu definieren und den Datenzugriff in Ihrem Data Lake von einer zentralen Stelle aus zu überprüfen. Sie können die detaillierte Zugriffskontrolle von Lake Formation verwenden, um Ihre vorhandenen Datenkatalogressourcen und Amazon S3 S3-Datenstandorte zu verwalten.

Mit Daten, die bei Lake Formation registriert sind, können Sie Datenkatalogressourcen sicher zwischen IAM-Prinzipalen, AWS Konten, AWS Organisationen und Organisationseinheiten gemeinsam nutzen.

Weitere Informationen zum Erstellen von Datenkatalogressourcen mit Lake Formation finden Sie unter Erstellen von Datenkatalogtabellen und Datenbanken im AWS Lake Formation Entwicklerhandbuch.

Amazon Athena

Amazon Athena verwendet den Datenkatalog, um Tabellenmetadaten für die Amazon S3 S3-Daten in Ihrem AWS Konto zu speichern und abzurufen. Mithilfe der Tabellenmetadaten kann die Athena-

Abfrage-Engine wissen, wie die Daten, die Sie abfragen möchten, gefunden, gelesen und verarbeitet werden.

Sie können die AWS Glue Data Catalog direkt mit `CREATE TABLE` Athena-Anweisungen auffüllen. Sie können das Schema und die Partitionsmetadaten im Datenkatalog manuell definieren und auffüllen, ohne einen Crawler ausführen zu müssen.

1. Erstellen Sie in der Athena-Konsole eine Datenbank, in der die Tabellenmetadaten im Datenkatalog gespeichert werden.
2. Verwenden Sie die `CREATE EXTERNAL TABLE` Anweisung, um das Schema Ihrer Datenquelle zu definieren.
3. Verwenden Sie die `PARTITIONED BY` Klausel, um alle Partitionsschlüssel zu definieren, wenn Ihre Daten partitioniert sind.
4. Verwenden Sie die `LOCATION` Klausel, um den Amazon S3 S3-Pfad anzugeben, in dem Ihre tatsächlichen Datendateien gespeichert werden.
5. Ausführen der `CREATE TABLE`-Anweisung.

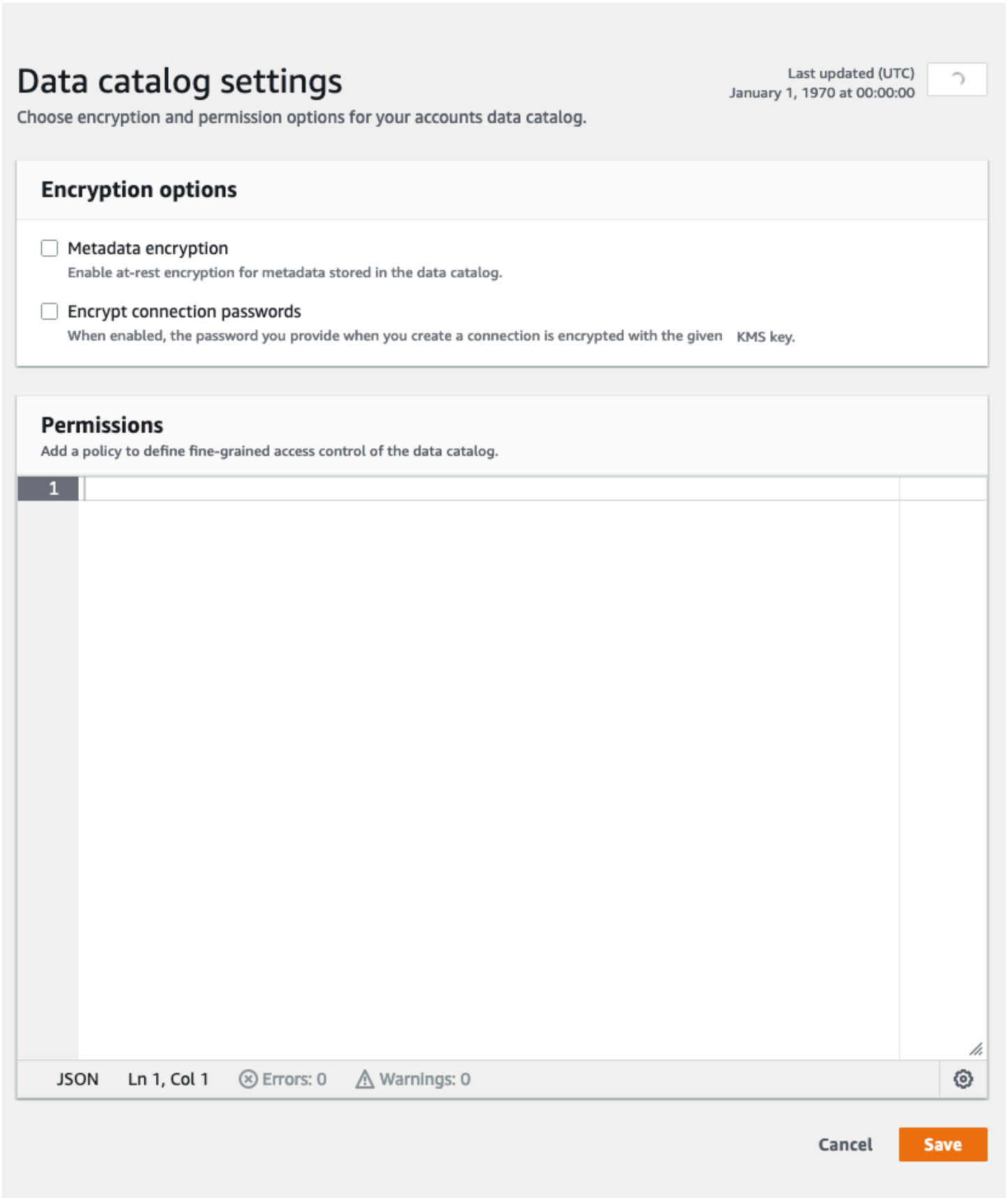
   Diese Abfrage erstellt die Tabellenmetadaten im Datenkatalog auf der Grundlage Ihres definierten Schemas und Ihrer Partitionen, ohne die Daten tatsächlich zu crawlen.

Sie können die Tabelle in Athena abfragen, und es verwendet die Metadaten aus dem Datenkatalog, um auf Ihre Datendateien in Amazon S3 zuzugreifen und diese abzufragen.

Weitere Informationen finden Sie unter Erstellen von Datenbanken und Tabellen im Amazon Athena Athena-Benutzerhandbuch.

Einstellungen für den Datenkatalog

Die Datenkatalogeinstellungen enthalten Optionen zum Festlegen von Verschlüsselungs- und Berechtigungsoptionen für den Datenkatalog in Ihrem Konto.


Data catalog settings
Choose encryption and permission options for your accounts data catalog.
Last updated (UTC)
January 1, 1970 at 00:00:00
Encryption options
Metadata encryption
Enable at-rest encryption for metadata stored in the data catalog.
Encrypt connection passwords
When enabled, the password you provide when you create a connection is encrypted with the given KMS key.
Permissions
Add a policy to define fine-grained access control of the data catalog.
1
JSON
Ln 1, Col 1
Errors: 0
Warnings: 0
Cancel
Save

Ändern der differenzierten Zugangskontrolle von Data Catalog

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie einen Verschlüsselungsoption aus.

   - Metadatenverschlüsselung – Aktivieren Sie dieses Kontrollkästchen zum Verschlüsseln der Metadaten in Ihrem Data Catalog. Metadaten werden im Ruhezustand mit dem von Ihnen angegebenen Schlüssel AWS Key Management Service (AWS KMS) verschlüsselt. Weitere Informationen finden Sie unter Verschlüsseln Ihres Data Catalog.
   - Verbindungskennwörter verschlüsseln — Aktivieren Sie dieses Kontrollkästchen, um Kennwörter in AWS Glue Verbindungsobjekt, wenn die Verbindung erstellt oder aktualisiert wird. Passwörter werden mit dem von Ihnen angegebenen AWS KMS Schlüssel verschlüsselt. Passwörter werden bei Rückgabe verschlüsselt. Diese Option ist eine globale Einstellung für alle AWS Glue Verbindungen im Datenkatalog. Wenn Sie dieses Kontrollkästchen deaktivieren, bleiben zuvor verschlüsselte Passwörter mit dem Schlüssel verschlüsselt, der bei Ihrer Erstellung oder Aktualisierung verwendet wurde. Weitere Informationen zur AWS Glue Verbindungen, sieheHerstellen einer Verbindung zu Daten.

     Wenn Sie diese Option aktivieren, wählen Sie einen AWS KMS Schlüssel oder wählen Sie Enter a key ARN und geben Sie den Amazon-Ressourcennamen (ARN) für den Schlüssel ein. Geben Sie den ARN in der Form `arn:aws:kms:region:account-id:key/key-id` ein. Sie können den ARN auch in Form eines Schlüssel-Alias bereitstellen, wie z. B. `arn:aws:kms:region:account-id:alias/alias-name` .

     > **Important**
     >
     > Wenn diese Option ausgewählt wird, muss jeder Benutzer oder jede Rolle, der bzw. die eine Verbindung erstellt oder aktualisiert, auf dem angegebenen KMS-Schlüssel über die `kms:Encrypt`-Berechtigung verfügen.

     Weitere Informationen finden Sie unter Verschlüsselung von Verbindungspasswörtern.
3. Wählen Sie Settings (Einstellungen) aus und fügen Sie dann im Editor für Permissions (Berechtigungen) die Richtlinienanweisung hinzu, um die differenzierte Zugriffskontrolle von Data Catalog für Ihr Konto ändern. Es kann jeweils nur eine Richtlinie gleichzeitig an einen Data Catalog angefügt werden. Sie können eine JSON-Ressourcenrichtlinie in dieses Steuerelement

einfügen. Weitere Informationen finden Sie unter Ressourcenbasierte Richtlinien innerhalb von Glue AWS.

4. Wählen Sie Save (Speichern) aus, um Ihren Data Catalog mit allen Änderungen, die Sie vorgenommen haben, zu aktualisieren.

Sie können auch verwenden AWS Glue API-Operationen zum Speichern, Abrufen und Löschen von Ressourcenrichtlinien. Weitere Informationen finden Sie unter Sicherheit APIs in AWS Glue.

Auffüllen und Verwalten von Transaktionstabellen

Apache Iceberg, Apache Hudi und Linux Foundation Delta Lake sind Open-Source-Tabellenformate, die für die Verarbeitung umfangreicher Datenanalysen und Data Lake-Workloads in Apache Spark entwickelt wurden.

Sie können Iceberg-, Hudi- und Delta Lake-Tabellen mit den folgenden Methoden füllen: AWS Glue Data Catalog

- AWS-Glue-Crawler; — AWS-Glue-Crawler s kann automatisch Iceberg-, Hudi- und Delta Lake-Tabellenmetadaten im Datenkatalog erkennen und auffüllen. Weitere Informationen finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs .
- AWS Glue ETL-Jobs — Sie können ETL-Jobs erstellen, um Daten in Iceberg-, Hudi- und Delta Lake-Tabellen zu schreiben und deren Metadaten im Datenkatalog aufzufüllen. Weitere Informationen finden Sie unter Verwenden von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.
- AWS Glue Konsole, AWS Lake Formation Konsole AWS CLI oder API — Sie können die Konsole, die Lake Formation AWS Glue Formation-Konsole oder die API verwenden, um Iceberg-Tabellendefinitionen im Datenkatalog zu erstellen und zu verwalten.

Themen

- Apache Iceberg-Tabellen erstellen
- Optimieren von Iceberg-Tabellen
- Optimierung der Abfrageleistung für Iceberg-Tabellen

Apache Iceberg-Tabellen erstellen

Sie können Apache Iceberg-Tabellen erstellen, die das Apache Parquet-Datenformat verwenden, AWS Glue Data Catalog wobei sich die Daten in Amazon S3 befinden. Eine Tabelle im Datenkatalog ist die Metadatendefinition, die die Daten in einem Datenspeicher darstellt. AWS Glue Erstellt standardmäßig Iceberg v2-Tabellen. Den Unterschied zwischen v1- und v2-Tabellen finden Sie unter Formatversionsänderungen in der Apache-Iceberg-Dokumentation.

Apache Iceberg ist ein offenes Tabellenformat für sehr große analytische Datensätze. Iceberg ermöglicht einfache Änderungen an Ihrem Schema, auch bekannt als Schemaentwicklung, was bedeutet, dass Benutzer Spalten zu einer Datentabelle hinzufügen, umbenennen oder daraus entfernen können, ohne die zugrunde liegenden Daten zu stören. Iceberg bietet auch Unterstützung für die Datenversionierung, sodass Benutzer Änderungen an Daten im Laufe der Zeit verfolgen können. Dadurch wird die Zeitreisefunktion aktiviert, mit der Benutzer auf historische Versionen von Daten zugreifen und diese abfragen und Datenänderungen zwischen Aktualisierungen und Löschungen analysieren können.

Sie können die Lake Formation Formation-Konsole oder den `CreateTable` Vorgang in der AWS Glue API verwenden AWS Glue , um eine Iceberg-Tabelle im Datenkatalog zu erstellen. Weitere Informationen finden Sie unter CreateTable action (Python: create_table).

Wenn Sie eine Iceberg-Tabelle im Datenkatalog erstellen, müssen Sie das Tabellenformat und den Metadatendateipfad in Amazon S3 angeben, um Lese- und Schreibvorgänge durchführen zu können.

Sie können Lake Formation verwenden, um Ihre Iceberg-Tabelle mithilfe detaillierter Zugriffskontrollberechtigungen zu sichern, wenn Sie den Amazon S3 S3-Datenstandort bei registrieren. AWS Lake Formation Für Quelldaten in Amazon S3 und Metadaten, die nicht bei Lake Formation registriert sind, wird der Zugriff durch IAM-Berechtigungsrichtlinien für Amazon S3 und AWS Glue Aktionen bestimmt. Weitere Informationen finden Sie unter Berechtigungen verwalten.

> **Note**
>
> Data Catalog unterstützt nicht das Erstellen von Partitionen und das Hinzufügen von Iceberg-Tabelleneigenschaften.

Voraussetzungen

Um Iceberg-Tabellen im Datenkatalog zu erstellen und Lake Formation Formation-Datenzugriffsberechtigungen einzurichten, müssen Sie die folgenden Anforderungen erfüllen:

1. Zum Erstellen von Iceberg-Tabellen ohne die bei Lake Formation registrierten Daten sind Berechtigungen erforderlich.

   Zusätzlich zu den Berechtigungen, die zum Erstellen einer Tabelle im Datenkatalog erforderlich sind, benötigt der Tabellenersteller die folgenden Berechtigungen:

   - `s3:PutObject`auf der Ressource arn:aws:s3::: {bucketName}
   - `s3:GetObject`auf der Ressource arn:aws:s3::: {bucketName}
   - `s3:DeleteObject`auf der Ressource arn:aws:s3::: {bucketName}
2. Erforderliche Berechtigungen zum Erstellen von Iceberg-Tabellen mit bei Lake Formation registrierten Daten:

   Um Lake Formation zur Verwaltung und Sicherung der Daten in Ihrem Data Lake zu verwenden, registrieren Sie Ihren Amazon S3 S3-Standort, der die Daten für Tabellen enthält, bei Lake Formation. Auf diese Weise kann Lake Formation Anmeldeinformationen an AWS Analysedienste wie Athena, Redshift Spectrum und Amazon EMR weitergeben, um auf Daten zuzugreifen. Weitere Informationen zur Registrierung eines Amazon S3 S3-Standorts finden Sie unter Hinzufügen eines Amazon S3 S3-Standorts zu Ihrem Data Lake.

   Ein Principal, der die zugrunde liegenden Daten liest und schreibt, die bei Lake Formation registriert sind, benötigt die folgenden Berechtigungen:

   - `lakeformation:GetDataAccess`
   - `DATA_LOCATION_ACCESS`

     Ein Principal, der über Datenspeicherberechtigungen für einen Standort verfügt, hat auch Standortberechtigungen für alle untergeordneten Standorte.

     Weitere Informationen zu Zugriffsberechtigungen für Daten finden Sie unter Zugrundeliegende Datenzugriffskontrolle (U-Link).

Um die Komprimierung zu aktivieren, muss der Dienst eine IAM-Rolle übernehmen, die über Berechtigungen zum Aktualisieren von Tabellen im Datenkatalog verfügt. Details hierzu finden Sie unter Voraussetzungen für die Tabellenoptimierung

Eine Iceberg-Tabelle erstellen

Sie können Iceberg v1- und v2-Tabellen mit AWS Glue Lake Formation Formation-Konsole oder AWS Command Line Interface wie auf dieser Seite dokumentiert erstellen. Sie können Iceberg-Tabellen auch mit dem erstellen. AWS-Glue-Crawler Weitere Informationen finden Sie unter Datenkatalog und Crawler im AWS Glue Entwicklerhandbuch.

Um eine Iceberg-Tabelle zu erstellen

Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie unter Datenkatalog die Option Tabellen aus und verwenden Sie die Schaltfläche Tabelle erstellen, um die folgenden Attribute anzugeben:
   - Tabellenname — Geben Sie einen Namen für die Tabelle ein. Wenn Sie Athena für den Zugriff auf Tabellen verwenden, verwenden Sie diese Benennungstipps im Amazon Athena Athena-Benutzerhandbuch.
   - Datenbank — Wählen Sie eine bestehende Datenbank aus oder erstellen Sie eine neue.
   - Beschreibung — Die Beschreibung der Tabelle. Sie können eine Beschreibung zum besseren Verständnis der Inhalte der Tabelle schreiben.
   - Tabellenformat — Wählen Sie als Tabellenformat Apache Iceberg.
   - Komprimierung aktivieren — Wählen Sie Komprimierung aktivieren, um kleine Amazon S3 S3-Objekte in der Tabelle zu größeren Objekten zu komprimieren.
   - IAM-Rolle — Um die Komprimierung auszuführen, übernimmt der Service in Ihrem Namen eine IAM-Rolle. Sie können über das Dropdown-Menü eine IAM-Rolle auswählen. Die Rolle sollte die erforderlichen Berechtigungen für die Verdichtung haben.

     Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter. Voraussetzungen für die Tabellenoptimierung
   - Speicherort — Geben Sie den Pfad zu dem Ordner in Amazon S3 an, in dem die Metadatentabelle gespeichert ist. Iceberg benötigt eine Metadatendatei und einen Speicherort im Datenkatalog, um Lese- und Schreibvorgänge durchführen zu können.

- Schema — Wählen Sie Spalten hinzufügen, um Spalten und Datentypen der Spalten hinzuzufügen. Sie haben die Möglichkeit, eine leere Tabelle zu erstellen und das Schema später zu aktualisieren. Der Datenkatalog unterstützt Hive-Datentypen. Weitere Informationen finden Sie unter Hive-Datentypen.

  Mit Iceberg können Sie Schema und Partition weiterentwickeln, nachdem Sie die Tabelle erstellt haben. Sie können Athena-Abfragen verwenden, um das Tabellenschema zu aktualisieren, und Spark-Abfragen, um Partitionen zu aktualisieren.

AWS CLI

```
aws glue create-table \
    --database-name iceberg-db \
    --region us-west-2 \
    --open-table-format-input '{
      "IcebergInput": {
           "MetadataOperation": "CREATE",
           "Version": "2"
        }
      }' \
    --table-input '{"Name":"test-iceberg-input-demo",
            "TableType": "EXTERNAL_TABLE",
            "StorageDescriptor":{
               "Columns":[
                   {"Name":"col1", "Type":"int"},
                   {"Name":"col2", "Type":"int"},
                   {"Name":"col3", "Type":"string"}
                ],
               "Location":"s3://DOC_EXAMPLE_BUCKET_ICEBERG/"
            }
       }'
```

Optimieren von Iceberg-Tabellen

AWS Glue unterstützt mehrere Optionen zur Tabellenoptimierung, um die Verwaltung und Leistung von Apache Iceberg-Tabellen zu verbessern, die von den Analyse-Engines und ETL-Jobs verwendet werden. AWS Diese Optimierer sorgen für eine effiziente Speichernutzung, eine verbesserte

Abfrageleistung und ein effektives Datenmanagement. Es gibt drei Arten von Tabellenoptimierern in: AWS Glue

- Komprimierung — Bei der Datenkomprimierung werden kleine Datendateien komprimiert, um den Speicherverbrauch zu reduzieren und die Leseleistung zu verbessern. Datendateien werden zusammengeführt und neu geschrieben, um veraltete Daten zu entfernen und fragmentierte Daten in größeren, effizienteren Dateien zu konsolidieren. Sie können die Komprimierung so konfigurieren, dass sie automatisch ausgeführt wird.

  Binpack ist die Standardkomprimierungsstrategie in Apache Iceberg. Es kombiniert kleinere Datendateien zu größeren, um eine optimale Leistung zu erzielen. Die Komprimierung unterstützt auch Sortier- und Z-Order-Strategien, bei denen ähnliche Daten gruppiert werden. Sort organisiert Daten auf der Grundlage bestimmter Spalten und verbessert so die Abfrageleistung bei gefilterten Vorgängen. Z-Order erstellt sortierte Datensätze, die die Abfrageleistung verbessern, wenn mehrere Spalten gleichzeitig abgefragt werden. Alle drei Komprimierungsstrategien — bincpak, sort und Z-order — reduzieren die Menge der von Abfrage-Engines gescannten Daten und senken so die Kosten für die Abfrageverarbeitung.
- Aufbewahrung von Snapshots — Snapshots sind Versionen einer Iceberg-Tabelle mit Zeitstempel. Mit Konfigurationen zur Aufbewahrung von Snapshots können Kunden festlegen, wie lange und wie viele Snapshots aufbewahrt werden sollen. Die Konfiguration eines Optimizers für die Aufbewahrung von Snapshots kann helfen, den Speicheraufwand zu minimieren, indem ältere, unnötige Snapshots und die zugehörigen zugrunde liegenden Dateien entfernt werden.
- Löschen verwaister Dateien — Verwaiste Dateien sind Dateien, auf die in den Metadaten der Iceberg-Tabelle nicht mehr verwiesen wird. Diese Dateien können sich im Laufe der Zeit ansammeln, insbesondere nach Vorgängen wie dem Löschen von Tabellen oder fehlgeschlagenen ETL-Jobs. Wenn Sie das Löschen verwaister Dateien aktivieren AWS Glue , können Sie diese unnötigen Dateien regelmäßig identifizieren und entfernen, wodurch Speicherplatz frei wird.

Sie können die Optimierer für die Komprimierung, die Aufbewahrung von Snapshots und das Löschen verwaister Dateien für einzelne Iceberg-Tabellen im Datenkatalog mithilfe der AWS Glue Konsole oder API-Operationen aktivieren oder deaktivieren. AWS CLI AWS Glue

Themen

- Voraussetzungen für die Tabellenoptimierung
- Optimierung der Verdichtung
- Optimierung der Snapshot-Aufbewahrung

- Löschen verwaister Dateien
- Optimierungsdetails anzeigen
- Amazon CloudWatch Metriken anzeigen
- Löschen eines Optimierers
- Überlegungen und Einschränkungen
- Unterstützte Regionen für Tabellenoptimierer

Voraussetzungen für die Tabellenoptimierung

Der Tabellenoptimierer übernimmt die Berechtigungen der Rolle AWS Identity and Access Management (IAM), die Sie angeben, wenn Sie Optimierungsoptionen (Komprimierung, Aufbewahrung von Snapshots und Löschen verwaister Dateien) für eine Tabelle aktivieren. Sie können entweder eine einzelne Rolle für alle Optimierer oder separate Rollen für jeden Optimierer erstellen.

> **Note**
>
> Für den Optimierer für das Löschen verwaister Dateien sind die Berechtigungen oder nicht erforderlich. `glue:updateTable` `s3:putObject` Die Optimierer für den Ablauf und die Komprimierung von Snapshots benötigen dieselben Berechtigungen.

Die IAM-Rolle muss die Berechtigungen zum Lesen von Daten und Aktualisieren von Metadaten im Datenkatalog haben. Sie können eine IAM-Rolle erstellen und die folgenden Inline-Richtlinien anfügen:

- Fügen Sie die folgende Inline-Richtlinie hinzu, die Amazon S3 read/write S3-Berechtigungen für den Standort für Daten gewährt, für die nicht registriert sind AWS Lake Formation. Diese Richtlinie umfasst auch Berechtigungen zum Aktualisieren der Tabelle im Datenkatalog und zum Hinzufügen von Protokollen AWS Glue zu Protokollen und zum Veröffentlichen von Metriken. Amazon CloudWatch Für Quelldaten in Amazon S3, die nicht bei Lake Formation registriert sind, wird der Zugriff durch IAM-Berechtigungsrichtlinien für Amazon-S3- und AWS Glue -Aktionen bestimmt.

  Ersetzen Sie `bucket-name` in den folgenden Inline-Richtlinien durch den Namen Ihres Amazon-S3-Buckets, `aws-account-id` und `region` durch eine gültige AWS -Kontonummer und Region

des Datenkatalogs, `database_name` durch den Namen Ihrer Datenbank und `table_name` durch den Namen der Tabelle.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:UpdateTable",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<aws-account-id>:table/<database-name>/<table-
name>",
                "arn:aws:glue:<region>:<aws-account-id>:database/<database-name>",
                "arn:aws:glue:<region>:<aws-account-id>:catalog"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
```

```
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
compaction/logs:*",
                "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
retention/logs:*",
                "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
orphan-file-deletion/logs:*"
        }
    ]
}
```

- Verwenden Sie die folgende Richtlinie, um die Verdichtung für Daten zu aktivieren, die bei Lake Formation registriert sind.

  Wenn der Optimierungsrolle keine `IAM_ALLOWED_PRINCIPALS` Gruppenberechtigungen für die Tabelle erteilt wurden, benötigt die Rolle Lake Formation`ALTER`,`DESCRIBE`, `INSERT` und `DELETE` Berechtigungen für die Tabelle.

  Weitere Informationen zur Registrierung eines Amazon S3 S3-Buckets bei Lake Formation finden Sie unter Hinzufügen eines Amazon S3 S3-Standorts zu Ihrem Data Lake.

```
{
 "Version": "2012-10-17",
 "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "lakeformation:GetDataAccess"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
        "glue:UpdateTable",
        "glue:GetTable"
        ],

        "Resource": [
```

```
            "arn:aws:glue:<region>:<aws-account-
id>:table/<databaseName>/<tableName>",
            "arn:aws:glue:<region>:<aws-account-id>:database/<database-name>",
            "arn:aws:glue:<region>:<aws-account-id>:catalog"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
compaction/logs:*",
            "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
retention/logs:*",
            "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/iceberg-
orphan-file-deletion/logs:*"
        }
    ]
 }
```

- (Optional) Um Iceberg-Tabellen mit Daten in Amazon S3 S3-Buckets zu optimieren, die mit serverseitiger Verschlüsselung verschlüsselt wurden, benötigt die Verdichtungsrolle Berechtigungen zum Entschlüsseln von Amazon S3 S3-Objekten und zum Generieren eines neuen Datenschlüssels, um Objekte in die verschlüsselten Buckets zu schreiben. Fügen Sie dem gewünschten Schlüssel die folgende Richtlinie hinzu. AWS KMS Wir unterstützen nur Verschlüsselung auf Bucket-Ebene.

```
{
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::<aws-account-id>:role/<optimizer-role-name>"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
```

```
}
```

- (Optional) Für den bei Lake Formation registrierten Datenspeicherort benötigt die Rolle, die zur Registrierung des Speicherorts verwendet wird, Berechtigungen zum Entschlüsseln von Amazon-S3-Objekten und Generieren eines neuen Datenschlüssels, um Objekte in die verschlüsselten Buckets zu schreiben. Weitere Informationen finden Sie unter Registrieren eines verschlüsselten Amazon-S3-Speicherorts.
- (Optional) Wenn der AWS KMS Schlüssel in einem anderen AWS Konto gespeichert ist, müssen Sie der Verdichtungsrolle die folgenden Berechtigungen hinzufügen.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:<REGION>:<KEY_OWNER_ACCOUNT_ID>:key/<KEY_ID>"
            ]
        }
    ]
}
```

- Die Rolle, die Sie zum Ausführen der Verdichtung verwenden, muss die `iam:PassRole`-Berechtigung für die Rolle haben.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
```

```
                "arn:aws:iam::<account-id>:role/<optimizer-role-name>"
            ]
        }
    ]
}
```

- Fügen Sie der Rolle die folgende Vertrauensrichtlinie hinzu, damit der AWS Glue Dienst die IAM-Rolle zur Ausführung des Verdichtungsprozesses übernimmt.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
 }
```

Optimierung der Verdichtung

Die Amazon S3 S3-Data Lakes, die offene Tabellenformate wie Apache Iceberg verwenden, speichern Daten als S3-Objekte. Tausende kleiner Amazon S3 S3-Objekte in einer Data-Lake-Tabelle erhöhen den Metadaten-Overhead und beeinträchtigen die Leseleistung. AWS Glue Data Catalog bietet verwaltete Komprimierung für Iceberg-Tabellen und komprimiert kleine Objekte zu größeren, um eine bessere Leseleistung durch AWS Analysedienste wie Amazon EMR Amazon Athena und ETL-Jobs zu erzielen. AWS Glue Data Catalog führt die Komprimierung durch, ohne gleichzeitige Abfragen zu beeinträchtigen, und unterstützt die Komprimierung nur für Tabellen im Parquet-Format.

Der Tabellenoptimierer überwacht kontinuierlich Tabellenpartitionen und startet den Komprimierungsprozess, wenn der Schwellenwert für die Anzahl der Dateien und Dateigrößen überschritten wird.

Im Datenkatalog beginnt der Komprimierungsprozess, wenn eine Tabelle oder eine ihrer Partitionen mehr als 100 Dateien enthält. Jede Datei muss kleiner als 75% der Zieldateigröße sein. Die Zieldateigröße wird durch die `write.target-file-size-bytes` Tabelleneigenschaft definiert, die standardmäßig 512 MB beträgt, sofern sie nicht explizit festgelegt ist.

Einschränkungen finden Sie unter Unterstützte Formate und Einschränkungen für die verwaltete Datenkomprimierung .

Themen

- Verdichtungsoptimierer aktivieren
- Deaktivieren des Verdichtungsoptimierers

Verdichtungsoptimierer aktivieren

Sie können die AWS Glue Konsole oder AWS API verwenden AWS CLI, um die Komprimierung für Ihre Apache Iceberg-Tabellen im Datenkatalog zu aktivieren. AWS Glue Für neue Tabellen können Sie Apache Iceberg als Tabellenformat auswählen und die Verdichtung beim Erstellen der Tabellen aktivieren. Für neue Tabellen ist die Verdichtung standardmäßig deaktiviert.

Console

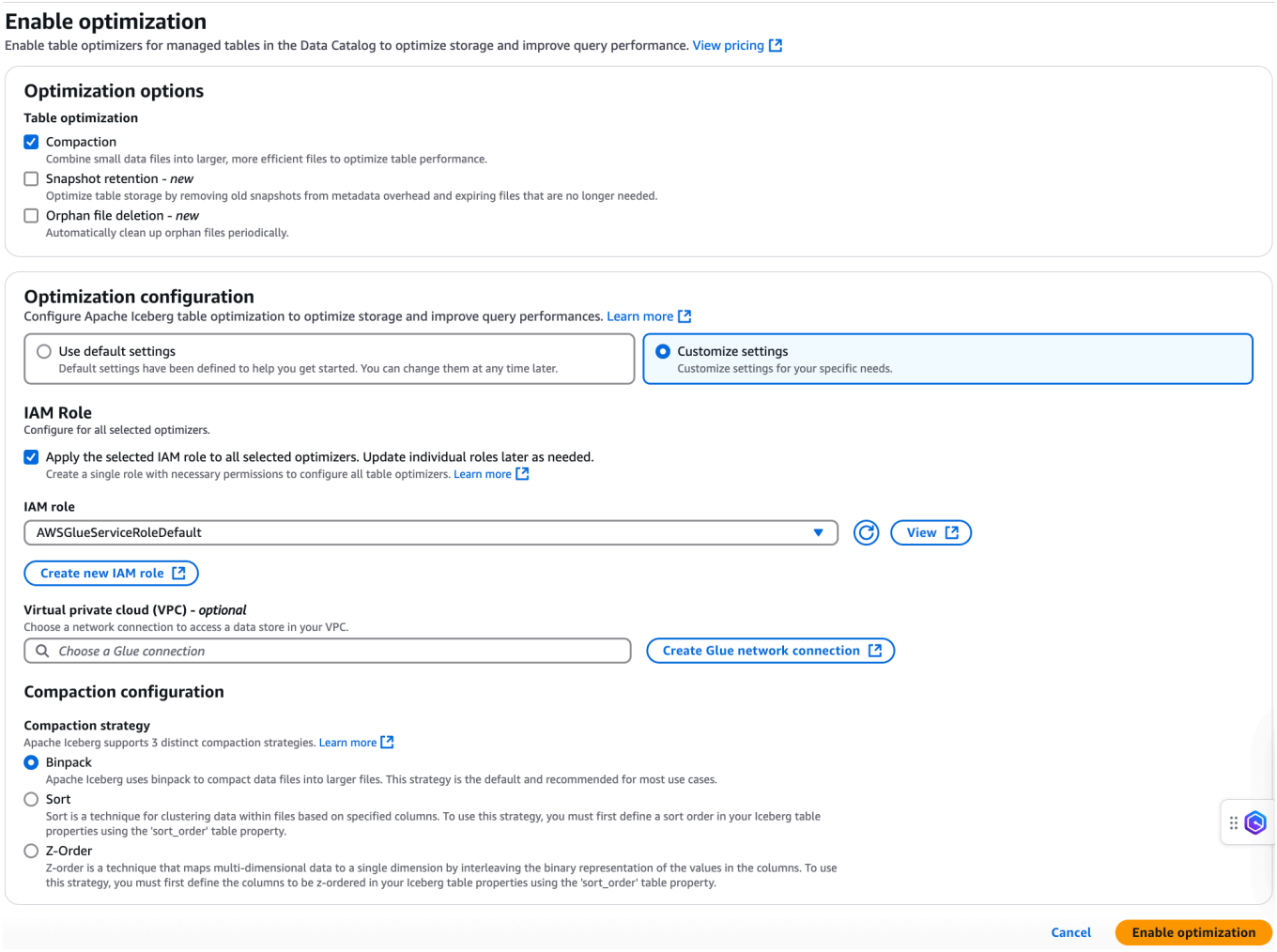
Aktivieren der Verdichtung

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/und melden Sie sich als Data Lake-Administrator, als Tabellenersteller oder als Benutzer an, dem die `lakeformation:GetDataAccess` Berechtigungen `glue:UpdateTable` und für die Tabelle erteilt wurden.
2. Wählen Sie im Navigationsbereich unter Datenkatalog die Option Tabellen aus.
3. Wählen Sie auf der Seite Tabellen eine Tabelle im geöffneten Tabellenformat aus, für die Sie die Komprimierung aktivieren möchten. Wählen Sie dann im Menü Aktionen die Option Optimierung und dann Aktivieren aus.

   Sie können die Komprimierung auch aktivieren, indem Sie auf der Seite mit den Tabellendetails die Registerkarte Tabellenoptimierung auswählen. Wählen Sie im unteren Bereich der Seite die Registerkarte Tabellenoptimierung und dann Verdichtung aktivieren aus.

   Die Option Optimierung aktivieren ist auch verfügbar, wenn Sie eine neue Iceberg-Tabelle im Datenkatalog erstellen.

4. Wählen Sie auf der Seite Optimierung aktivieren unter Optimierungsoptionen die Option Komprimierung aus.



5. Wählen Sie als Nächstes eine IAM-Rolle aus der Drop-down-Liste mit den im Abschnitt angegebenen Berechtigungen aus. Voraussetzungen für die Tabellenoptimierung

   Sie können auch die Option Neue IAM-Rolle erstellen auswählen, um eine benutzerdefinierte Rolle mit den erforderlichen Berechtigungen zum Ausführen der Komprimierung zu erstellen.

   Gehen Sie wie folgt vor, um eine vorhandene IAM-Rolle zu aktualisieren:

   a. Um die Berechtigungsrichtlinie für die IAM-Rolle zu aktualisieren, wechseln Sie in der IAM-Konsole zu der IAM-Rolle, die zum Ausführen der Verdichtung verwendet wird.

   b. Wählen Sie im Abschnitt Berechtigungen hinzufügen die Option Richtlinie erstellen aus. Erstellen Sie im neu geöffneten Browserfenster eine neue Richtlinie, die Sie mit Ihrer Rolle verwenden möchten.

c. Wählen Sie auf der Seite Richtlinie erstellen die `JSON` Registerkarte aus. Kopieren Sie den in den Voraussetzungen angezeigten JSON-Code in das Feld Richtlinien-Editor.

6. Wenn Sie Sicherheitsrichtlinien-Konfigurationen haben, bei denen der Iceberg-Tabellenoptimierer von einer bestimmten Virtual Private Cloud (VPC) aus auf Amazon S3 S3-Buckets zugreifen muss, stellen Sie eine AWS Glue Netzwerkverbindung her oder verwenden Sie eine bestehende.

   Wenn Sie noch keine AWS Glue VPC-Verbindung eingerichtet haben, erstellen Sie eine neue, indem Sie die Schritte im Abschnitt Verbindungen für Konnektoren erstellen mithilfe der AWS Glue Konsole oder des AWS CLI/SDKs ausführen.

7. Wählen Sie eine Verdichtungsstrategie. Die verfügbaren Optionen sind:

   - Binpack — Binpack ist die Standard-Komprimierungsstrategie in Apache Iceberg. Sie kombiniert kleinere Datendateien zu größeren, um eine optimale Leistung zu erzielen.
   - Sortieren — Bei der Sortierung in Apache Iceberg handelt es sich um eine Datenorganisationstechnik, bei der Informationen innerhalb von Dateien auf der Grundlage bestimmter Spalten gruppiert werden. Dadurch wird die Abfrageleistung erheblich verbessert, da die Anzahl der zu verarbeitenden Dateien reduziert wird. Sie definieren die Sortierreihenfolge in den Metadaten von Iceberg mithilfe des Sortierreihenfolgenfeldes. Wenn mehrere Spalten angegeben sind, werden die Daten in der Reihenfolge sortiert, in der die Spalten in der Sortierreihenfolge erscheinen, sodass Datensätze mit ähnlichen Werten zusammen in Dateien gespeichert werden. Mit der Strategie zur Sortierung und Komprimierung wird die Optimierung noch weiter vorangetrieben, indem Daten in allen Dateien innerhalb einer Partition sortiert werden.
   - Z-Reihenfolge — Die Z-Reihenfolge ist eine Möglichkeit, Daten zu organisieren, wenn Sie nach mehreren Spalten mit gleicher Wichtigkeit sortieren müssen. Im Gegensatz zur herkömmlichen Sortierung, bei der eine Spalte gegenüber anderen priorisiert wird, verleiht die Z-Reihenfolge jeder Spalte ein ausgewogenes Gewicht, sodass Ihre Abfrage-Engine bei der Suche nach Daten weniger Dateien liest.

     Bei dieser Technik werden die Binärziffern von Werten aus verschiedenen Spalten zusammengefügt. Wenn Sie beispielsweise die Zahlen 3 und 4 aus zwei Spalten haben, werden sie bei der Z-Reihenfolge zuerst in Binärzahlen umgewandelt (3 wird zu 011 und 4 wird zu 100). Dann werden diese Ziffern verschachtelt, um einen neuen Wert zu erstellen: 011010. Durch diese Verschachtelung entsteht ein Muster, das zusammengehörige Daten physisch nahe beieinander hält.

Die Z-Reihenfolge ist besonders effektiv bei mehrdimensionalen Abfragen. Beispielsweise kann eine Kundentabelle, die nach Einkommen, Bundesland und Postleitzahl Z-sortiert ist, im Vergleich zur hierarchischen Sortierung eine bessere Leistung bieten, wenn Abfragen über mehrere Dimensionen hinweg durchgeführt werden. Diese Organisation ermöglicht Abfragen, die auf bestimmte Kombinationen aus Einkommen und geografischem Standort abzielen, um schnell relevante Daten zu finden und gleichzeitig unnötige Dateiscans zu minimieren.

8. Wählen Sie Optimierung aktivieren.

AWS CLI

Im folgenden Beispiel wird gezeigt, wie Sie die Verdichtung aktivieren. Ersetzen Sie die Konto-ID durch eine gültige AWS Konto-ID. Ersetzen Sie den Datenbanknamen und den Tabellennamen durch die tatsächlichen Tabellen- und Datenbanknamen in Iceberg. Ersetzen Sie den `roleArn` durch den AWS -Ressourcennamen (ARN) der IAM-Rolle und den Namen der IAM-Rolle, die über die erforderlichen Berechtigungen zum Ausführen der Verdichtung verfügt. Sie können die Verdichtungsstrategie `sort` durch andere unterstützte Strategien wie `z-order` oder `binpack` ersetzen.

bestellen“ je nach Ihren Anforderungen.

```
aws glue create-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration '{
    "roleArn": "arn:aws:iam::123456789012:role/optimizer_role",
    "enabled": true,
    "vpcConfiguration": {"glueConnectionName": "glue_connection_name"},
    "compactionConfiguration": {
      "icebergConfiguration": {"strategy": "sort"}
    }
  }'\
--type compaction
```

AWS API

Rufen Sie die Operation CreateTableOptimizer auf, um die Verdichtung für eine Tabelle zu aktivieren.

Nachdem Sie die Komprimierung aktiviert haben, werden auf der Registerkarte „Tabellenoptimierung“ die folgenden Verdichtungsdetails angezeigt, sobald der Verdichtungslauf abgeschlossen ist:

Startzeit

Der Zeitpunkt, zu dem der Verdichtungsvorgang im Datenkatalog gestartet wurde. Der Wert ist ein Zeitstempel in UTC-Zeit.

Endzeit

Der Zeitpunkt, zu dem der Verdichtungsvorgang in Data Catalog endete. Der Wert ist ein Zeitstempel in UTC-Zeit.

Status

Der Status des Verdichtungslaufs. Die Werte sind „Erfolgreich“ oder „Fehlgeschlagen“.

Komprimierte Dateien

Gesamtzahl der komprimierten Dateien.

Komprimierte Bytes

Gesamtzahl der komprimierten Byte.

Deaktivieren des Verdichtungsoptimierers

Sie können die automatische Komprimierung für eine bestimmte Apache Iceberg-Tabelle mithilfe der Konsole oder deaktivieren. AWS Glue AWS CLI

Console

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/
2. Wählen Sie in der linken Navigationsleiste unter Datenkatalog die Option Tabellen aus.
3. Wählen Sie aus der Tabellenliste die Iceberg-Tabelle aus, für die Sie die Komprimierung deaktivieren möchten.
4. Wählen Sie im unteren Bereich der Tabellendetailseite den Tab Tabellenoptimierung.
5. Wählen Sie unter Aktionen die Option Deaktivieren und anschließend Komprimierung aus.
6. Klicken Sie in der Bestätigungsmeldung auf Verdichtung deaktivieren. Sie können die Verdichtung später wieder aktivieren.

Nachdem Sie die Deaktivierung bestätigt haben, wird die Verdichtung deaktiviert und der Verdichtungsstatus für die Tabelle wird wieder auf `Disabled` gesetzt.

AWS CLI

Ersetzen Sie im folgenden Beispiel die Konto-ID durch eine gültige AWS Konto-ID. Ersetzen Sie den Datenbanknamen und den Tabellennamen durch die tatsächlichen Tabellen- und Datenbanknamen in Iceberg. Ersetzen Sie das `roleArn` durch den AWS Ressourcennamen (ARN) der IAM-Rolle und den tatsächlichen Namen der IAM-Rolle, die über die erforderlichen Berechtigungen zum Ausführen der Komprimierung verfügt.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "enabled":'false',
 "vpcConfiguration":{"glueConnectionName":"glue_connection_name"}}'\
  --type compaction
```

AWS API

Rufen Sie die Operation UpdateTableOptimizer auf, um die Verdichtung für eine bestimmte Tabelle zu deaktivieren.

Optimierung der Snapshot-Aufbewahrung

Mit der Snapshot-Aufbewahrungsfunktion von Apache Iceberg können Benutzer historische Daten zu bestimmten Zeitpunkten abfragen und unerwünschte Änderungen an ihren Tabellen rückgängig machen. Im AWS Glue Datenkatalog steuert die Konfiguration der Snapshot-Aufbewahrung, wie lange diese Snapshots (Versionen der Tabellendaten) aufbewahrt werden, bevor sie ablaufen und entfernt werden. Dies hilft bei der Verwaltung der Speicherkosten und des Metadaten-Overheads, indem ältere Snapshots auf der Grundlage einer konfigurierten Aufbewahrungszeit oder der maximalen Anzahl von aufzubewahrenden Snapshots automatisch entfernt werden.

Sie können den Aufbewahrungszeitraum in Tagen und die maximale Anzahl von Snapshots, die für eine Tabelle aufbewahrt werden sollen, konfigurieren. AWS Glue entfernt Snapshots, die älter als der angegebene Aufbewahrungszeitraum sind, aus den Tabellenmetadaten, wobei die neuesten

Snapshots bis zum konfigurierten Limit beibehalten werden. AWS Glue Löscht nach dem Entfernen alter Snapshots aus den Metadaten die entsprechenden Daten und Metadatendateien, auf die nicht mehr verwiesen wird und die für die abgelaufenen Snapshots eindeutig sind. Auf diese Weise können Zeitreiseabfragen nur bis zu den verbleibenden gespeicherten Snapshots abgerufen werden, während gleichzeitig Speicherplatz zurückgewonnen wird, der für abgelaufene Snapshot-Daten verwendet wurde.

Themen

- Snapshot Retention Optimizer aktivieren
- Aktualisierung des Snapshot Retention Optimizers
- Snapshot Retention Optimizer deaktivieren

Snapshot Retention Optimizer aktivieren

Sie können die AWS Glue Konsole oder die AWS API verwenden AWS CLI, um die Snapshot-Aufbewahrungsoptimierung für Ihre Apache Iceberg-Tabellen im Datenkatalog zu aktivieren. Für neue Tabellen können Sie Apache Iceberg als Tabellenformat wählen und den Snapshot Retention Optimizer aktivieren, wenn Sie die Tabelle erstellen. Die Aufbewahrung von Snapshots ist für neue Tabellen standardmäßig deaktiviert.

Console

Um den Snapshot Retention Optimizer zu aktivieren

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/und melden Sie sich als Data Lake-Administrator, als Tabellenersteller oder als Benutzer an, dem die `lakeformation:GetDataAccess` Berechtigungen `glue:UpdateTable` und für die Tabelle erteilt wurden.
2. Wählen Sie im Navigationsbereich unter Datenkatalog die Option Tabellen aus.
3. Wählen Sie auf der Seite Tabellen eine Iceberg-Tabelle aus, für die Sie den Snapshot Retention Optimizer aktivieren möchten, und wählen Sie dann im Menü Aktionen unter Optimierung die Option Aktivieren aus.

   Sie können die Optimierung auch aktivieren, indem Sie die Tabelle auswählen und die Seite mit den Tabellendetails öffnen. Wählen Sie im unteren Bereich der Seite die Registerkarte Tabellenoptimierung und wählen Sie Snapshot-Aufbewahrung aktivieren aus.

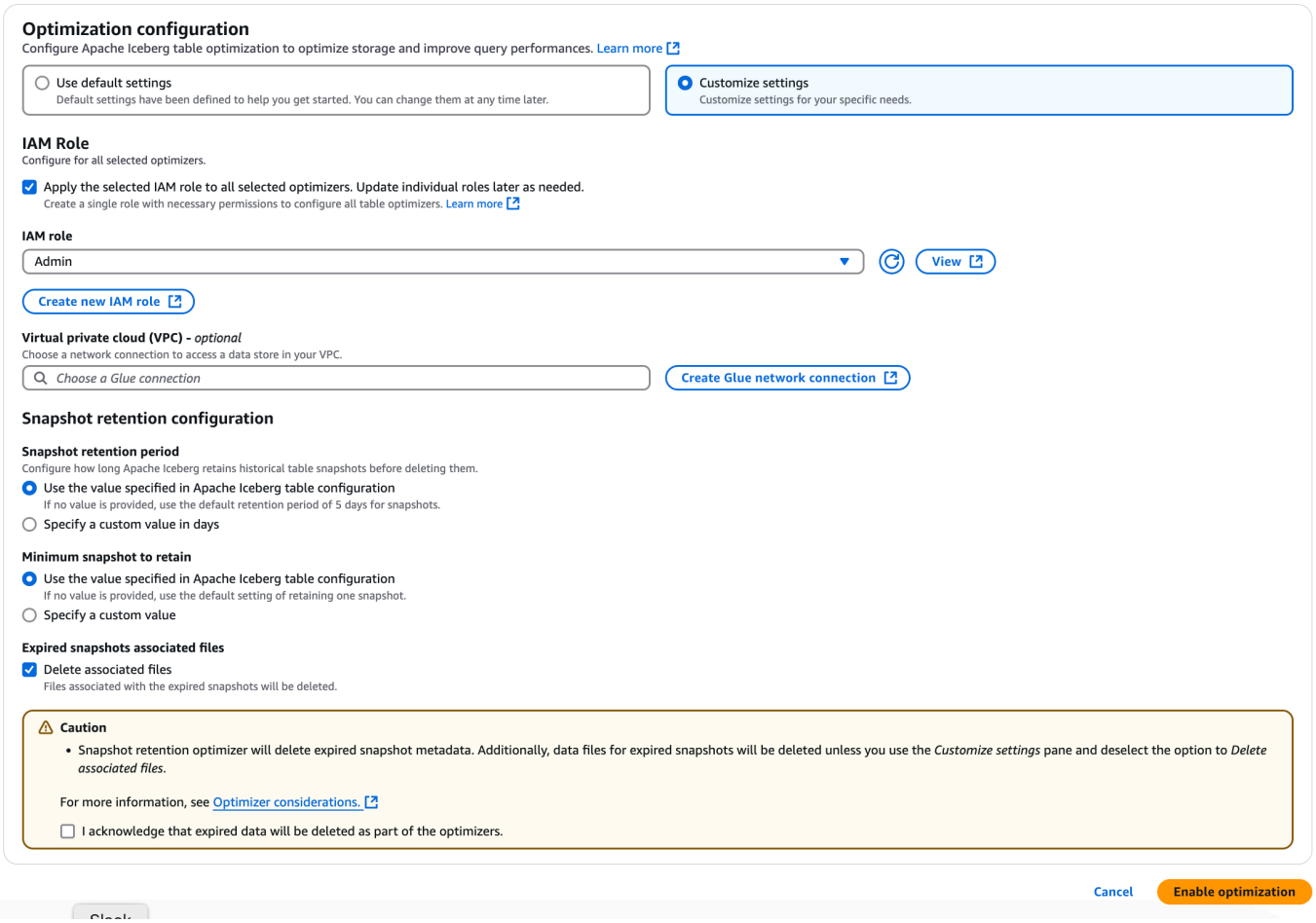
4. Auf der Seite Optimierung aktivieren haben Sie unter Optimierungskonfiguration zwei Optionen: Standardeinstellung verwenden oder Einstellungen anpassen. Wenn Sie sich dafür entscheiden, die Standardeinstellungen zu verwenden, AWS Glue verwendet die in der Iceberg-Tabellenkonfiguration definierten Eigenschaften, um den Aufbewahrungszeitraum für Snapshots und die Anzahl der beizubehaltenden Snapshots zu bestimmen. Wenn diese Konfiguration nicht vorhanden ist, wird ein Snapshot fünf Tage lang AWS Glue aufbewahrt und die mit den abgelaufenen Snapshots verknüpften Dateien gelöscht.
5. Wählen Sie als Nächstes eine IAM-Rolle aus, die in Ihrem Namen die Ausführung des Optimierers übernehmen AWS Glue kann. Einzelheiten zu den für die IAM-Rolle erforderlichen Berechtigungen finden Sie im Abschnitt. Voraussetzungen für die Tabellenoptimierung

   Gehen Sie wie folgt vor, um eine vorhandene IAM-Rolle zu aktualisieren:

   a. Um die Berechtigungsrichtlinie für die IAM-Rolle zu aktualisieren, wechseln Sie in der IAM-Konsole zu der IAM-Rolle, die zum Ausführen der Verdichtung verwendet wird.

   b. Wählen Sie im Abschnitt Berechtigungen hinzufügen die Option Richtlinie erstellen aus. Erstellen Sie im neu geöffneten Browserfenster eine neue Richtlinie, die Sie mit Ihrer Rolle verwenden möchten.

   c. Wählen Sie auf der Seite Richtlinie erstellen die Registerkarte JSON aus. Kopieren Sie den in den Voraussetzungen angezeigten JSON-Code in das Feld Richtlinien-Editor.
6. Wenn Sie die Werte für die Snapshot-Aufbewahrungskonfiguration lieber manuell festlegen möchten, wählen Sie Einstellungen anpassen.



7. Wählen Sie das Kästchen Ausgewählte IAM-Rolle auf die ausgewählten Optimierer anwenden, um eine einzige IAM-Rolle für alle zu verwenden und alle Optimierer zu aktivieren.
8. Wenn Sie Sicherheitsrichtlinien-Konfigurationen haben, bei denen der Iceberg-Tabellenoptimierer von einer bestimmten Virtual Private Cloud (VPC) aus auf Amazon S3 S3-Buckets zugreifen muss, stellen Sie eine AWS Glue Netzwerkverbindung her oder verwenden Sie eine bestehende.

   Wenn Sie noch keine AWS Glue VPC-Verbindung eingerichtet haben, erstellen Sie eine neue, indem Sie die Schritte im Abschnitt Verbindungen für Konnektoren erstellen mithilfe der AWS Glue Konsole oder des AWS CLI/SDKs ausführen.
9. Wählen Sie als Nächstes unter Snapshot-Aufbewahrungskonfiguration entweder die in der Iceberg-Tabellenkonfiguration angegebenen Werte aus, oder geben Sie benutzerdefinierte Werte für die Aufbewahrungsdauer von Snapshots an (history.expire). max-snapshot-age-ms) und die Mindestanzahl von Snapshots (history.expire. min-snapshots-to-keep) beizubehalten.
10. Wählen Sie „Zugeordnete Dateien löschen“, um die zugrunde liegenden Dateien zu löschen, wenn der Tabellenoptimierer alte Schnappschüsse aus den Tabellenmetadaten löscht.

Wenn Sie diese Option nicht wählen und ältere Snapshots aus den Tabellenmetadaten entfernt werden, verbleiben die zugehörigen Dateien als verwaiste Dateien im Speicher.

11. Lesen Sie als Nächstes den Warnhinweis und wählen Sie Ich bestätige, um fortzufahren.

> **Note**
>
> Im Datenkatalog berücksichtigt der Snapshot Retention Optimizer den Lebenszyklus, der durch Aufbewahrungsrichtlinien auf Branchen- und Tag-Ebene gesteuert wird. Weitere Informationen finden Sie im Abschnitt Branching and Tagging in der Iceberg-Dokumentation.

12. Überprüfen Sie die Konfiguration und wählen Sie Optimierung aktivieren.

Warten Sie einige Minuten, bis der Retention Optimizer ausgeführt wird und alte Snapshots, die auf der Konfiguration basieren, ablaufen lassen.

AWS CLI

Um die Aufbewahrung von Snapshots für neue Iceberg-Tabellen in zu aktivieren AWS Glue, müssen Sie einen Tabellenoptimierer des Typs erstellen `retention` und das `enabled` Feld auf in setzen. `true` `table-optimizer-configuration` Sie können dies mit dem AWS CLI Befehl `create-table-optimizer` oder tun. `update-table-optimizer` Darüber hinaus müssen Sie die Felder für die Aufbewahrungskonfiguration entsprechend Ihren Anforderungen angeben. `snapshotRetentionPeriodInDays` `numberOfSnapshotsToRetain`

Das folgende Beispiel zeigt, wie Sie den Snapshot Retention Optimizer aktivieren. Ersetzen Sie die Konto-ID durch eine gültige AWS Konto-ID. Ersetzen Sie den Datenbanknamen und den Tabellennamen durch die tatsächlichen Tabellen- und Datenbanknamen in Iceberg. Ersetzen Sie das `roleArn` durch den AWS Ressourcennamen (ARN) der IAM-Rolle und den Namen der IAM-Rolle, die über die erforderlichen Berechtigungen verfügt, um den Snapshot Retention Optimizer auszuführen.

```
aws glue create-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
```

```
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":'true',
 "vpcConfiguration":{
"glueConnectionName":"glue_connection_name"}, "retentionConfiguration":
{"icebergConfiguration":
{"snapshotRetentionPeriodInDays":7,"numberOfSnapshotsToRetain":3,"cleanExpiredFiles":'true'}
  --type retention
```

Mit diesem Befehl wird ein Aufbewahrungsoptimierer für die angegebene Iceberg-Tabelle im angegebenen Katalog, in der angegebenen Datenbank und Region erstellt. Der table-optimizer-configuration spezifiziert den zu verwendenden ARN der IAM-Rolle, aktiviert den Optimierer und legt die Aufbewahrungskonfiguration fest. In diesem Beispiel werden Snapshots 7 Tage lang aufbewahrt, mindestens 3 Snapshots werden aufbewahrt und abgelaufene Dateien werden bereinigt.

- snapshotRetentionPeriodInDays — Die Anzahl der Tage, für die Snapshots aufbewahrt werden sollen, bevor sie ablaufen. Der Standardwert ist 5.
- numberOfSnapshotsToRetain — Die Mindestanzahl von Snapshots, die aufbewahrt werden müssen, auch wenn sie älter als die Aufbewahrungsfrist sind. Der Standardwert ist 1.
- cleanExpiredFiles — Ein boolescher Wert, der angibt, ob abgelaufene Datendateien nach ablaufenden Snapshots gelöscht werden sollen. Der Standardwert ist `true`.

  Wenn dieser Wert auf true gesetzt ist, werden ältere Snapshots aus den Tabellenmetadaten entfernt und die ihnen zugrunde liegenden Dateien werden gelöscht. Wenn dieser Parameter auf false gesetzt ist, werden ältere Snapshots aus den Tabellenmetadaten entfernt, aber die ihnen zugrunde liegenden Dateien verbleiben als verwaiste Dateien im Speicher.

AWS API

Rufen Sie CreateTableOptimizerden Vorgang auf, um den Snapshot Retention Optimizer für eine Tabelle zu aktivieren.

Nachdem Sie die Verdichtung aktiviert haben, werden auf der Registerkarte Tabellenoptimierung die folgenden Verdichtungsdetails angezeigt (nach etwa 15 bis 20 Minuten):

Startzeit

Der Zeitpunkt, zu dem der Snapshot Retention Optimizer gestartet wurde. Der Wert ist ein Zeitstempel in UTC-Zeit.

Run time (Laufzeit)

Die Zeit gibt an, wie lange der Optimierer benötigt, um die Aufgabe abzuschließen. Der Wert ist ein Zeitstempel in UTC-Zeit.

Status

Der Status der Ausführung des Optimierers. Die Werte sind „Erfolgreich“ oder „Fehlgeschlagen“.

Datendateien wurden gelöscht

Gesamtzahl der gelöschten Dateien.

Manifestdateien wurden gelöscht

Gesamtzahl der gelöschten Manifestdateien.

Manifestlisten wurden gelöscht

Gesamtzahl der gelöschten Manifestlisten.

Aktualisierung des Snapshot Retention Optimizers

Sie können die bestehende Konfiguration eines Snapshot-Aufbewahrungsoptimierers für eine bestimmte Apache Iceberg-Tabelle mithilfe der AWS Glue Konsole oder der API AWS CLI aktualisieren. UpdateTableOptimizer

Console

Um die Konfiguration der Snapshot-Aufbewahrung zu aktualisieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie Datenkatalog und dann Tabellen aus. Wählen Sie aus der Tabellenliste die Iceberg-Tabelle aus, für die Sie die Snapshot Retention Optimizer-Konfiguration aktualisieren möchten.
3. Wählen Sie im unteren Bereich der Tabellendetailseite die Registerkarte Tabellenoptimierung und dann Bearbeiten aus. Sie können auch im Aktionsmenü in der oberen rechten Ecke der Seite die Option Bearbeiten unter Optimierung auswählen.

4. Nehmen Sie auf der Seite Optimierung bearbeiten die gewünschten Änderungen vor.
5. Wählen Sie Save (Speichern) aus.

AWS CLI

Um einen Snapshot-Aufbewahrungsoptimierer mit dem zu aktualisieren AWS CLI, können Sie den folgenden Befehl verwenden:

```
aws glue update-table-optimizer \
 --catalog-id 123456789012 \
 --database-name iceberg_db \
 --table-name iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role"","enabled":'true',
 "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"},"retentionConfiguration":
{"icebergConfiguration":
{"snapshotRetentionPeriodInDays":7,"numberOfSnapshotsToRetain":3,"cleanExpiredFiles":'true'}
 \
 --type retention
```

Dieser Befehl aktualisiert die Aufbewahrungskonfiguration für die angegebene Tabelle im angegebenen Katalog, in der angegebenen Datenbank und Region. Die wichtigsten Parameter sind:

- snapshotRetentionPeriodInDays — Die Anzahl der Tage, für die Snapshots aufbewahrt werden sollen, bevor sie ablaufen. Der Standardwert ist 1.
- numberOfSnapshotsToRetain — Die Mindestanzahl von Snapshots, die aufbewahrt werden müssen, auch wenn sie älter als die Aufbewahrungsfrist sind. Der Standardwert ist 5.
- cleanExpiredFiles — Ein boolescher Wert, der angibt, ob abgelaufene Datendateien nach ablaufenden Snapshots gelöscht werden sollen. Der Standardwert ist `true`.

  Wenn dieser Wert auf true gesetzt ist, werden ältere Snapshots aus den Tabellenmetadaten entfernt und die ihnen zugrunde liegenden Dateien werden gelöscht.“ Wenn dieser Parameter auf „false“ gesetzt ist, werden ältere Snapshots aus den Tabellenmetadaten entfernt, die ihnen zugrunde liegenden Dateien verbleiben jedoch als verwaiste Dateien im Speicher.

API

Um einen Tabellenoptimierer zu aktualisieren, können Sie die API verwenden. `UpdateTableOptimizer` Mit dieser API können Sie die Konfiguration eines vorhandenen Tabellenoptimierers für die Komprimierung, Aufbewahrung oder Entfernung verwaister Dateien aktualisieren. Zu den Anforderungsparametern gehören:

- catalogId (erforderlich): Die ID des Katalogs, der die Tabelle enthält
- databaseName (optional): Der Name der Datenbank, die die Tabelle enthält
- tableName (optional): Der Name der Tabelle
- type (erforderlich): Der Typ des Tabellenoptimierers (Komprimierung, Aufbewahrung oder orphan_file_deletion)
- RetentionConfiguration (erforderlich): Die aktualisierte Konfiguration für den Tabellenoptimierer, einschließlich Rollen-ARN, aktiviertem Status, Aufbewahrungskonfiguration und Konfiguration zum Entfernen verwaister Dateien.

Snapshot Retention Optimizer deaktivieren

Sie können den Snapshot Retention Optimizer für eine bestimmte Apache Iceberg-Tabelle mithilfe der Konsole oder deaktivieren. AWS Glue AWS CLI

Console

Um die Aufbewahrung von Snapshots zu deaktivieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie Datenkatalog und dann Tabellen aus. Wählen Sie aus der Tabellenliste die Iceberg-Tabelle aus, für die Sie den Optimizer für die Aufbewahrung von Snapshots deaktivieren möchten.
3. Wählen Sie im unteren Bereich der Tabellendetailseite unter Aktionen die Option Tabellenoptimierung und Deaktivieren, Snapshot-Aufbewahrung aus.

   Sie können auch im Aktionsmenü in der oberen rechten Ecke der Seite die Option Deaktivieren unter Optimierung auswählen.
4. Wählen Sie in der Bestätigungsnachricht die Option Deaktivieren aus. Sie können den Snapshot Retention Optimizer zu einem späteren Zeitpunkt wieder aktivieren.

Nachdem Sie dies bestätigt haben, ist der Snapshot Retention Optimizer deaktiviert und der Status für die Snapshot-Aufbewahrung wird wieder auf gesetzt. `Not enabled`

AWS CLI

Ersetzen Sie im folgenden Beispiel die Konto-ID durch eine gültige AWS Konto-ID. Ersetzen Sie den Datenbanknamen und den Tabellennamen durch die tatsächlichen Tabellen- und Datenbanknamen in Iceberg. Ersetzen Sie das `roleArn` durch den AWS Ressourcennamen (ARN) der IAM-Rolle und den tatsächlichen Namen der IAM-Rolle, die über die erforderlichen Berechtigungen zum Ausführen des Aufbewahrungsoptimierers verfügt.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"}, "enabled":'false'}'\
  --type retention
```

AWS API

Rufen Sie UpdateTableOptimizerden Vorgang auf, um den Snapshot-Aufbewahrungsoptimizer für eine bestimmte Tabelle zu deaktivieren.

Löschen verwaister Dateien

AWS Glue Data Catalog ermöglicht es Ihnen, verwaiste Dateien aus Ihren Iceberg-Tabellen zu entfernen. Verwaiste Dateien sind Dateien, die in Ihrer Amazon S3 S3-Datenquelle unter dem angegebenen Tabellenspeicherort vorhanden sind, nicht von den Iceberg-Tabellenmetadaten verfolgt werden und älter sind als Ihre konfigurierte Altersbeschränkung. Diese verwaisten Dateien können sich im Laufe der Zeit aufgrund von Vorgängen wie Komprimieren, Löschen von Partitionen oder Neuschreiben von Tabellen ansammeln und unnötigen Speicherplatz beanspruchen.

Der Optimierer für das Löschen verwaister Dateien AWS Glue scannt die Tabellenmetadaten und die eigentlichen Datendateien, identifiziert die verwaisten Dateien und löscht sie, um Speicherplatz zurückzugewinnen.

Sie können das Löschen verwaister Dateien initiieren, indem Sie im Datenkatalog einen Tabellenoptimierer für das Löschen verwaister Dateien erstellen.

> **Important**
>
> Standardmäßig werden beim Löschen verwaister Dateien Dateien an Ihrem gesamten Tabellenspeicherort ausgewertet. AWS Glue Sie können zwar ein Unterpräfix konfigurieren, um den Umfang der Auswertung einzuschränken, müssen jedoch sicherstellen, dass Ihr Tabellenspeicherort keine Dateien aus anderen Datenquellen oder Tabellen enthält. Wenn sich der Speicherort Ihrer Tabelle mit anderen Datenquellen überschneidet, identifiziert und löscht der Dienst möglicherweise Dateien, die nichts miteinander zu tun haben, als verwaiste Dateien.

Themen

- Löschen verwaister Dateien aktivieren
- Der Optimizer für das Löschen verwaister Dateien wird aktualisiert
- Löschen verwaister Dateien deaktivieren

Löschen verwaister Dateien aktivieren

Sie können die AWS Glue Konsole oder die AWS API verwenden AWS CLI, um das Löschen verwaister Dateien für Ihre Apache Iceberg-Tabellen im Datenkatalog zu aktivieren. Für neue Tabellen können Sie Apache Iceberg als Tabellenformat wählen und den Optimizer für das Löschen verwaister Dateien aktivieren, wenn Sie die Tabelle erstellen. Die Aufbewahrung von Snapshots ist für neue Tabellen standardmäßig deaktiviert.

Console

Um das Löschen verwaister Dateien zu aktivieren

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/und melden Sie sich als Data Lake-Administrator, als Tabellenersteller oder als Benutzer an, dem die `lakeformation:GetDataAccess` Berechtigungen `glue:UpdateTable` und für die Tabelle erteilt wurden.
2. Wählen Sie im Navigationsbereich unter Datenkatalog die Option Tabellen aus.

3. Wählen Sie auf der Seite Tabellen eine Iceberg-Tabelle aus, in der Sie das Löschen verwaister Dateien aktivieren möchten.

   Wählen Sie im unteren Bereich der Seite die Registerkarte Tabellenoptimierung und dann unter Aktionen die Option Aktivieren, Löschen verwaister Dateien aus.

   Sie können auch im Aktionsmenü in der oberen rechten Ecke der Seite die Option Aktivieren unter Optimierung auswählen.
4. Wählen Sie auf der Seite Optimierung aktivieren unter Optimierungsoptionen die Option Löschen verwaister Dateien aus.
5. Wenn Sie die Standardeinstellungen verwenden, werden alle verwaisten Dateien nach 3 Tagen gelöscht. Wenn Sie die verwaisten Dateien für eine bestimmte Anzahl von Tagen behalten möchten, wählen Sie Einstellungen anpassen.
6. Wählen Sie als Nächstes eine IAM-Rolle mit den erforderlichen Berechtigungen zum Löschen verwaister Dateien aus.
7. Wenn Sie Sicherheitsrichtlinien-Konfigurationen haben, bei denen der Iceberg-Tabellenoptimierer von einer bestimmten Virtual Private Cloud (VPC) aus auf Amazon S3 S3-Buckets zugreifen muss, stellen Sie eine AWS Glue Netzwerkverbindung her oder verwenden Sie eine bestehende.

   Wenn Sie noch keine AWS Glue VPC-Verbindung eingerichtet haben, erstellen Sie eine neue, indem Sie die Schritte im Abschnitt Verbindungen für Konnektoren erstellen mithilfe der AWS Glue Konsole oder des AWS CLI/SDKs ausführen.
8. Wenn Sie Einstellungen anpassen wählen, geben Sie unter Konfiguration zum Löschen verwaister Dateien die Anzahl der Tage ein, für die die Dateien vor dem Löschen aufbewahrt werden sollen.
9. Wählen Sie Optimierung aktivieren aus.

AWS CLI

Um das Löschen verwaister Dateien für eine Iceberg-Tabelle in zu aktivieren AWS Glue, müssen Sie einen Tabellenoptimierer vom Typ 1 erstellen `orphan_file_deletion` und das `enabled` Feld auf „true“ setzen. Um mit dem einen Optimizer für das Löschen verwaister Dateien für eine Iceberg-Tabelle zu erstellen AWS CLI, können Sie den folgenden Befehl verwenden:

```
aws glue create-table-optimizer \
 --catalog-id 123456789012 \
```

```
 --database-name iceberg_db \
 --table-name iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":true,
 "vpcConfiguration":{
"glueConnectionName":"glue_connection_name"}, "orphanFileDeletionConfiguration":
{"icebergConfiguration":{"orphanFileRetentionPeriodInDays":3, "location":'S3
 location'}}}'\
 --type orphan_file_deletion
```

Dieser Befehl erstellt einen Optimierer für das Löschen verwaister Dateien für die angegebene Iceberg-Tabelle. Die wichtigsten Parameter sind:

- roleArn — der ARN der IAM-Rolle mit Berechtigungen für den Zugriff auf den S3-Bucket und die Glue-Ressourcen.
- enabled — Auf true setzen, um den Optimizer zu aktivieren.
- orphanFileRetentionPeriodInDays — Die Anzahl der Tage, für die verwaiste Dateien aufbewahrt werden sollen, bevor sie gelöscht werden (mindestens 1 Tag).
- type — Auf orphan_file_deletion setzen, um einen Optimierer für das Löschen verwaister Dateien zu erstellen.

Nach der Erstellung des Tabellenoptimierers führt er in regelmäßigen Abständen das Löschen verwaister Dateien durch (einmal täglich, sofern aktiviert). Sie können die Läufe mithilfe der `list-table-optimizer-runs` API überprüfen. Der Job zum Löschen verwaister Dateien identifiziert und löscht Dateien, die nicht in den Iceberg-Metadaten für die Tabelle erfasst sind.

API

Rufen Sie den CreateTableOptimizerVorgang auf, um den Optimizer für das Löschen verwaister Dateien für eine bestimmte Tabelle zu erstellen.

Der Optimizer für das Löschen verwaister Dateien wird aktualisiert

Sie können die Konfiguration für den Optimizer für das Löschen verwaister Dateien ändern, z. B. den Aufbewahrungszeitraum für verwaiste Dateien oder die vom Optimierer verwendete IAM-Rolle mithilfe AWS Glue der Konsole oder des Vorgangs ändern. AWS CLI`UpdateTableOptimizer`

AWS Management Console

Um den Optimizer für das Löschen verwaister Dateien zu aktualisieren

1. Wählen Sie Datenkatalog und dann Tabellen aus. Wählen Sie aus der Tabellenliste die Tabelle aus, für die Sie die Konfiguration des Optimierers für das Löschen verwaister Dateien aktualisieren möchten.
2. Wählen Sie im unteren Bereich der Tabellendetailseite die Option Tabellenoptimierung und dann Bearbeiten aus.
3. Nehmen Sie auf der Seite Optimierung bearbeiten die gewünschten Änderungen vor.
4. Wählen Sie Save (Speichern) aus.

AWS CLI

Sie können den `update-table-optimizer` Aufruf verwenden, um den Optimizer für das Löschen verwaister Dateien zu aktualisieren. Sie können den Aufruf verwenden. AWS Glue Auf diese Weise können Sie `OrphanFileDeletionConfiguration` in dem `icebergConfiguration` Feld, in dem Sie den aktualisierten `OrphanFileRetentionPeriodInDays` Wert angeben können, die Anzahl der Tage für die Aufbewahrung verwaister Dateien festlegen und den Speicherort der Iceberg-Tabelle angeben, aus dem verwaiste Dateien gelöscht werden sollen.

```
aws glue update-table-optimizer \
 --catalog-id 123456789012 \
 --database-name iceberg_db \
 --table-name Iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":true,
 "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"},"orphanFileDeletionConfiguration":
{"icebergConfiguration":{"orphanFileRetentionPeriodInDays":5}}}' \
 --type orphan_file_deletion
```

API

Rufen Sie den UpdateTableOptimizerVorgang auf, um den Optimizer für das Löschen verwaister Dateien für eine Tabelle zu aktualisieren.

Löschen verwaister Dateien deaktivieren

Sie können den Optimizer für das Löschen verwaister Dateien für eine bestimmte Apache Iceberg-Tabelle mithilfe der Konsole oder deaktivieren. AWS Glue AWS CLI

Console

Um das Löschen verwaister Dateien zu deaktivieren

1. Wählen Sie Datenkatalog und dann Tabellen aus. Wählen Sie aus der Tabellenliste die Iceberg-Tabelle aus, für die Sie den Optimizer für das Löschen verwaister Dateien deaktivieren möchten.
2. Wählen Sie im unteren Bereich der Seite mit den Tabellendetails die Registerkarte Tabellenoptimierung aus.
3. Wählen Sie „Aktionen“ und anschließend „Deaktivieren“, „Löschen verwaister Dateien“.

   Sie können auch im Menü Aktionen unter Optimierung die Option Deaktivieren auswählen.
4. Wählen Sie in der Bestätigungsnachricht die Option Deaktivieren aus. Sie können den Optimizer für das Löschen verwaister Dateien zu einem späteren Zeitpunkt wieder aktivieren.

   Nachdem Sie dies bestätigt haben, ist der Optimizer für das Löschen verwaister Dateien deaktiviert und der Status für das Löschen verwaister Dateien wird wieder auf gelöscht. `Not enabled`

AWS CLI

Ersetzen Sie im folgenden Beispiel die Konto-ID durch eine gültige AWS Konto-ID. Ersetzen Sie den Datenbanknamen und den Tabellennamen durch die tatsächlichen Tabellen- und Datenbanknamen in Iceberg. Ersetzen Sie das `roleArn` durch den AWS Ressourcennamen (ARN) der IAM-Rolle und den tatsächlichen Namen der IAM-Rolle, die über die erforderlichen Berechtigungen zum Deaktivieren des Optimierers verfügt.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "enabled":'false'}'\
  --type orphan_file_deletion
```

API

Rufen Sie den UpdateTableOptimizerVorgang auf, um den Snapshot-Aufbewahrungsoptimizer für eine bestimmte Tabelle zu deaktivieren.

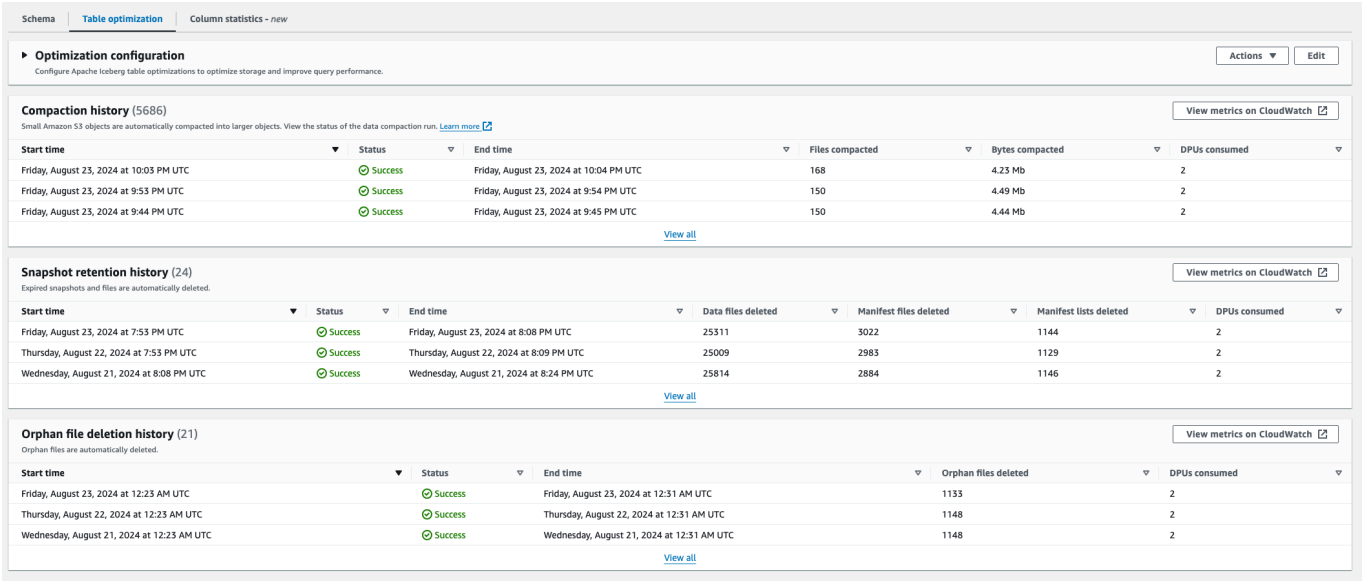
Optimierungsdetails anzeigen

Sie können den Optimierungsstatus für Apache Iceberg-Tabellen in der AWS Glue Konsole oder mithilfe von AWS API-Operationen anzeigen. AWS CLI

Console

Um den Optimierungsstatus für Iceberg-Tabellen (Konsole) anzuzeigen

- Sie können den Optimierungsstatus für Iceberg-Tabellen auf der AWS Glue Konsole anzeigen, indem Sie eine Iceberg-Tabelle aus der Tabellenliste unter Datenkatalog auswählen. Unter Tabellenoptimierung. Wählen Sie Alle anzeigen



AWS CLI

Sie können die Optimierungsdetails mit anzeigen. AWS CLI

Ersetzen Sie in den folgenden Beispielen die Konto-ID durch eine gültige AWS Konto-ID, den Datenbanknamen und den Tabellennamen durch den tatsächlichen Iceberg-Tabellennamen. Geben Sie für `type` einen Optimierungstyp an. Akzeptable Werte sind `compaction`, `retention` und `orphan-file-deletion`.

- Abrufen von Details der letzten Verdichtungsausführung für eine Tabelle

```
aws get-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- Verwenden Sie das folgende Beispiel, um den Verlauf eines Optimierers für eine bestimmte Tabelle abzurufen.

```
aws list-table-optimizer-runs \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- Das folgende Beispiel zeigt, wie der Optimierungslauf und die Konfigurationsdetails für mehrere Optimierer abgerufen werden. Sie können maximal 20 Optimierer angeben.

```
aws glue batch-get-table-optimizer \
--entries '[{"catalogId":"123456789012", "databaseName":"iceberg_db",
 "tableName":"iceberg_table", "type":"compaction"}]'
```

API

- Verwenden Sie die Operation `GetTableOptimizer`, um die Details der letzten Ausführung eines Optimierers abzurufen.
- Verwenden Sie die Operation `ListTableOptimizerRuns`, um den Verlauf eines bestimmten Optimierers für eine bestimmte Tabelle abzurufen. Sie können 20 Optimierer in einem einzigen API-Aufruf angeben.
- Verwenden Sie den BatchGetTableOptimizerVorgang, um Konfigurationsdetails für mehrere Optimierer in Ihrem Konto abzurufen.

Amazon CloudWatch Metriken anzeigen

Nach erfolgreicher Ausführung der Tabellenoptimierer erstellt der Service Amazon CloudWatch Metriken zur Leistung des Optimierungsjobs. Sie können zu den CloudWatch Metriken gehen und

Metriken, Alle Metriken auswählen. Sie können Metriken nach dem spezifischen Namespace (z. B. AWS Glue), dem Tabellennamen oder dem Datenbanknamen filtern.

Weitere Informationen finden Sie unter Anzeigen der verfügbaren Metriken im Benutzerhandbuch für Amazon CloudWatch .

Verdichtung

- Anzahl der verdichteten Byte
- Anzahl der verdichteten Dateien
- Anzahl der DPU, die dem Auftrag zugewiesen sind
- Auftragsdauer (Stunden)

Snapshot-Aufbewahrung

- Anzahl der gelöschten Datendateien
- Anzahl der gelöschten Manifestdateien
- Anzahl der gelöschten Manifestlisten
- Auftragsdauer (Stunden)

Löschen verwaister Dateien

- Anzahl der gelöschten verwaisten Dateien
- Auftragsdauer (Stunden)

Löschen eines Optimierers

Sie können einen Optimierer und die zugehörigen Metadaten für die Tabelle mithilfe AWS CLI unserer AWS API-Operation löschen.

Führen Sie den folgenden AWS CLI Befehl aus, um den Optimierungsverlauf für eine Tabelle zu löschen. Sie müssen den Optimizer `type` zusammen mit der Katalog-ID, dem Datenbanknamen und dem Tabellennamen angeben. Die zulässigen Werte sind: `compactionretention`, `undorphan_file_deletion`.

```
aws glue delete-table-optimizer \
```

```
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

Verwenden Sie die Operation `DeleteTableOptimizer`, um einen Optimierer für eine Tabelle zu löschen.

Überlegungen und Einschränkungen

Dieser Abschnitt enthält Dinge, die Sie bei der Verwendung von Tabellenoptimierern innerhalb von beachten sollten. AWS Glue Data Catalog

Unterstützte Formate und Einschränkungen für die verwaltete Datenkomprimierung

Die Datenkomprimierung unterstützt eine Vielzahl von Datentypen und Komprimierungsformaten zum Lesen und Schreiben von Daten, einschließlich des Lesens von Daten aus verschlüsselten Tabellen.

Die Datenverdichtung unterstützt:

- Verschlüsselung — Die Datenkomprimierung unterstützt nur die standardmäßige Amazon S3 S3-Verschlüsselung (SSE-S3) und die serverseitige KMS-Verschlüsselung (SSE-KMS).
- Verdichtungsstrategien — Binpack, Sortierung und Sortierung in Z-Reihenfolge
- Sie können die Verdichtung über das Konto ausführen, in dem sich der Datenkatalog befindet, wenn sich der Amazon-S3-Bucket, in dem die zugrunde liegenden Daten gespeichert werden, in einem anderen Konto befindet. Dazu benötigt die Verdichtungsrolle Zugriff auf den Amazon-S3-Bucket.

Die Datenverdichtung unterstützt derzeit nicht:

- Komprimierung für kontenübergreifende Tabellen — Sie können die Komprimierung nicht für kontenübergreifende Tabellen ausführen.
- Komprimierung für regionsübergreifende Tabellen — Sie können die Komprimierung nicht für regionsübergreifende Tabellen ausführen.
- Aktivieren der Verdichtung für Ressourcenlinks
- Tabellen in der Amazon S3 Express One Zone-Speicherklasse — Sie können keine Komprimierung auf S3 Express One Zone Iceberg-Tabellen ausführen.
- Die Z-Order-Komprimierungsstrategie unterstützt die folgenden Datentypen nicht:

- Dezimal
- TimestampWithoutZone

Überlegungen zu Optimierern für die Aufbewahrung von Snapshots und das Löschen verwaister Dateien

Die folgenden Überlegungen gelten für die Snapshot-Aufbewahrung und die Optimierer für das Löschen verwaister Dateien.

- Für die Prozesse zur Aufbewahrung von Snapshots und zum Löschen verwaister Dateien gilt eine Höchstgrenze von 1.000.000 Dateien pro Lauf. Wenn beim Löschen abgelaufener Snapshots die Anzahl der zum Löschen in Frage kommenden Dateien 1.000.000 übersteigt, werden alle verbleibenden Dateien, die diesen Schwellenwert überschreiten, weiterhin als verwaiste Dateien im Tabellenspeicher gespeichert.
- Snapshots werden vom Snapshot Retention Optimizer nur aufbewahrt, wenn beide Kriterien erfüllt sind: die Mindestanzahl der aufzubewahrenden Snapshots und der angegebene Aufbewahrungszeitraum.
- Der Snapshot Retention Optimizer löscht abgelaufene Snapshot-Metadaten aus Apache Iceberg, wodurch Zeitreiseanfragen für abgelaufene Snapshots verhindert und optional zugehörige Datendateien gelöscht werden.
- Der Optimizer für das Löschen verwaister Dateien löscht verwaiste Daten und Metadatendateien, auf die Iceberg-Metadaten nicht mehr verweisen, wenn ihre Erstellungszeit vor dem Aufbewahrungszeitraum für das Löschen verwaister Dateien ab dem Zeitpunkt der Ausführung des Optimizers liegt.
- Apache Iceberg erleichtert die Versionskontrolle durch Branches und Tags, die als Zeiger auf bestimmte Snapshot-Zustände bezeichnet werden. Jeder Branch und jedes Tag folgt einem eigenen unabhängigen Lebenszyklus, der durch Aufbewahrungsrichtlinien geregelt wird, die auf den jeweiligen Ebenen definiert sind. Die AWS Glue Data Catalog Optimierer berücksichtigen diese Lebenszyklusrichtlinien und stellen so die Einhaltung der festgelegten Aufbewahrungsregeln sicher. Aufbewahrungsrichtlinien auf Branchen- und Tagebene haben Vorrang vor den Optimierer-Konfigurationen.

  Weitere Informationen finden Sie unter Branching and Tagging in der Apache Iceberg-Dokumentation.
- Die Optimierer für die Aufbewahrung von Snapshots und das Löschen verwaister Dateien löschen Dateien, die gemäß den konfigurierten Parametern bereinigt werden können.

Verbessern Sie Ihre Kontrolle über das Löschen von Dateien, indem Sie S3-Versionierungs- und Lebenszyklusrichtlinien für die entsprechenden Buckets implementieren.

Ausführliche Anweisungen zum Einrichten der Versionierung und zum Erstellen von Lebenszyklusregeln finden Sie unter. https://docs.aws.amazon.com/AmazonS3/latest/userguide/Versioning.html

- Stellen Sie zur korrekten Bestimmung verwaister Dateien sicher, dass sich der angegebene Speicherort der Tabelle und alle Unterpfade nicht mit anderen Tabellen oder Datenquellen überschneiden oder Daten aus anderen Tabellen oder Datenquellen enthalten. Wenn sich Pfade überschneiden, riskieren Sie einen unwiederbringlichen Datenverlust durch unbeabsichtigtes Löschen von Dateien.

Unterstützte Regionen für Tabellenoptimierer

Die Funktionen zur Tabellenoptimierung (Komprimierung, Aufbewahrung von Snapshots und Löschen verwaister Dateien) für AWS Glue Data Catalog sind in den folgenden Bereichen verfügbar: AWS-Regionen

- Asien-Pazifik (Tokio)
- Asien-Pazifik (Seoul)
- Asia Pacific (Mumbai)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Kanada (Zentral)
- Europa (Irland)
- Europa (London)
- Europa (Frankfurt)
- Europa (Stockholm)
- USA Ost (Nord-Virginia)
- USA Ost (Ohio)
- USA West (Oregon)
- Südamerika (São Paulo)

Optimierung der Abfrageleistung für Iceberg-Tabellen

Apache Iceberg ist ein leistungsstarkes Open-Table-Format für riesige analytische Datensätze. AWS Glue unterstützt die Berechnung und Aktualisierung der Anzahl unterschiedlicher Werte (NDVs) für jede Spalte in Iceberg-Tabellen. Diese Statistiken können Dateningenieuren und Wissenschaftlern, die mit großen Datensätzen arbeiten, eine bessere Abfrageoptimierung, ein besseres Datenmanagement und eine bessere Leistungseffizienz ermöglichen.

AWS Glue schätzt die Anzahl der unterschiedlichen Werte in jeder Spalte der Iceberg-Tabelle und speichert sie in Puffin-Dateien auf Amazon S3, die mit Iceberg-Tabellen-Snapshots verknüpft sind. Puffin ist ein Iceberg-Dateiformat, das zum Speichern von Metadaten wie Indizes, Statistiken und Skizzen entwickelt wurde. Das Speichern von Skizzen in Puffin-Dateien, die mit Schnappschüssen verknüpft sind, gewährleistet die Transaktionskonsistenz und Aktualität der NDV-Statistiken.

Sie können konfigurieren, dass die Aufgabe zur Generierung von Spaltenstatistiken über die Konsole oder ausgeführt wird. AWS Glue AWS CLI Wenn Sie den Prozess starten, AWS Glue wird ein Spark-Job im Hintergrund gestartet und die AWS Glue Tabellenmetadaten im Datenkatalog aktualisiert. Sie können Spaltenstatistiken über die AWS Glue Konsole AWS CLI oder durch Aufrufen des GetColumnStatisticsForTableAPI-Vorgangs anzeigen.

> **Note**
>
> Wenn Sie AWS Lake Formation Berechtigungen verwenden, um den Zugriff auf die Tabelle zu steuern, erfordert die Rolle, die die Aufgabe für die Spaltenstatistik einnimmt, vollständigen Tabellenzugriff, um Statistiken zu generieren.

Themen

- Voraussetzungen für die Generierung von Spaltenstatistiken
- Generieren von Spaltenstatistiken für Iceberg-Tabellen
- Weitere Informationen finden Sie auch unter

Voraussetzungen für die Generierung von Spaltenstatistiken

Um Spaltenstatistiken für Iceberg-Tabellen zu generieren oder zu aktualisieren, übernimmt die Statistikgenerierungsaufgabe in Ihrem AWS Identity and Access Management Namen eine (IAM-)

Rolle. Basierend auf den der Rolle erteilten Berechtigungen kann die Aufgabe zur Generierung von Spaltenstatistiken die Daten aus dem Amazon-S3-Datenspeicher lesen.

Wenn Sie die Aufgabe zur Generierung von Spaltenstatistiken konfigurieren, AWS Glue können Sie eine Rolle erstellen, die die `AWSGlueServiceRole` AWS verwaltete Richtlinie sowie die erforderliche Inline-Richtlinie für die angegebene Datenquelle umfasst.

Wenn Sie eine bestehende Rolle für die Generierung von Spaltenstatistiken angeben, stellen Sie sicher, dass sie die `AWSGlueServiceRole` Richtlinie oder eine gleichwertige (oder eine Version mit eingeschränktem Geltungsbereich) und die erforderlichen Inline-Richtlinien enthält.

Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter Voraussetzungen für die Generierung von Spaltenstatistiken.

Generieren von Spaltenstatistiken für Iceberg-Tabellen

Gehen Sie wie folgt vor, um einen Zeitplan für die Generierung von Statistiken im Datenkatalog zu konfigurieren. Verwenden Sie dazu die AWS Glue Konsole AWS CLI oder führen Sie den StartColumnStatisticsTaskRunVorgang aus.

Um Spaltenstatistiken zu generieren

1. Melden Sie sich bei der AWS Glue Konsole an unter https://console.aws.amazon.com/glue/.
2. Wählen Sie unter Datenkatalog die Option Tabellen aus.
3. Wählen Sie eine Iceberg-Tabelle aus der Liste aus.
4. Wählen Sie im Menü Aktionen die Option Spaltenstatistiken, Auf Abruf generieren aus.

   Sie können auch im unteren Bereich der Seite Tabellen auf der Registerkarte Spaltenstatistiken die Schaltfläche Statistik generieren auswählen.
5. Geben Sie auf der Seite Statistiken generieren die Details zur Statistikgenerierung an. Folgen Sie den Schritten 6 bis 11 in Generieren von Spaltenstatistiken nach einem Zeitplan diesem Abschnitt, um einen Zeitplan für die Statistikgenerierung für Iceberg-Tabellen zu konfigurieren.

   Sie können sich auch dafür entscheiden, Spaltenstatistiken bei Bedarf zu generieren, indem Sie den Anweisungen in der Generierung von Spaltenstatistiken auf Abruf

   > **Note**
   >
   > Die Sampling-Option ist für Iceberg-Tabellen nicht verfügbar.

AWS Glue berechnet die Anzahl der unterschiedlichen Werte für jede Spalte der Iceberg-Tabelle in einer neuen Puffin-Datei, die an die angegebene Snapshot-ID an Ihrem Amazon S3-Standort übergeben wurde.

Weitere Informationen finden Sie auch unter

- Anzeigen von Spaltenstatistiken
- Anzeigen von Spaltenstatistik-Aufgabenläufen
- Beenden des Spaltenstatistik-Aufgabenlaufs
- Löschen von Spaltenstatistiken

Verwaltung des Datenkatalogs

Das AWS Glue Data Catalog ist ein zentrales Metadaten-Repository, das strukturelle und betriebliche Metadaten für Ihre Amazon S3 S3-Datensätze speichert. Die effektive Verwaltung des Datenkatalogs ist entscheidend für die Aufrechterhaltung der Datenqualität, Leistung, Sicherheit und Verwaltung.

Wenn Sie diese Methoden zur Verwaltung von Datenkatalogen verstehen und anwenden, können Sie sicherstellen, dass Ihre Metadaten korrekt, leistungsfähig, sicher und gut verwaltet bleiben, während sich Ihre Datenlandschaft weiterentwickelt.

In diesem Abschnitt werden die folgenden Aspekte der Datenkatalogverwaltung behandelt:

- Aktualisierung des Tabellenschemas und der Partitionen Während sich Ihre Daten weiterentwickeln, müssen Sie möglicherweise das im Datenkatalog definierte Tabellenschema oder die Partitionsstruktur aktualisieren. Weitere Informationen zur programmgesteuerten Durchführung dieser Aktualisierungen mithilfe von AWS Glue ETL finden Sie unter. Aktualisierung des Schemas und Hinzufügen neuer Partitionen im Datenkatalog mit AWS Glue ETL-Aufträge
- Verwaltung von Spaltenstatistiken: Präzise Spaltenstatistiken helfen dabei, Abfragepläne zu optimieren und die Leistung zu verbessern. Weitere Informationen zum Generieren, Aktualisieren und Verwalten von Spaltenstatistiken finden Sie unterOptimieren der Abfrageleistung mithilfe von Spaltenstatistiken.
- Den Datenkatalog verschlüsseln Um sensible Metadaten zu schützen, können Sie Ihren Datenkatalog mit AWS Key Management Service ()AWS KMS verschlüsseln. In diesem Abschnitt wird erklärt, wie Sie die Verschlüsselung für Ihren Datenkatalog aktivieren und verwalten.

- Die Sicherung des Datenkatalogs mit AWS Lake Formation Lake Formation bietet einen umfassenden Ansatz für die Sicherheit und Zugriffskontrolle von Data Lakes. Sie können Lake Formation verwenden, um den Zugriff auf Ihren Datenkatalog und die zugrunde liegenden Daten zu sichern und zu steuern.

Themen

- Aktualisierung des Schemas und Hinzufügen neuer Partitionen im Datenkatalog mit AWS Glue ETL-Aufträge
- Optimieren der Abfrageleistung mithilfe von Spaltenstatistiken
- Verschlüsseln Ihres Data Catalog
- Sicherung Ihres Datenkatalogs mit Lake Formation
- Arbeiten mit AWS Glue Data Catalog Ansichten in AWS Glue

Aktualisierung des Schemas und Hinzufügen neuer Partitionen im Datenkatalog mit AWS Glue ETL-Aufträge

Der ETL-Auftrag zum Extrahieren, Transformieren oder Laden kann neue Tabellenpartitionen im Zieldatenspeicher erstellen. Ihr Datensatz-Schema kann sich weiterentwickeln und von dem abweichen AWS Glue Datenkatalogschema im Laufe der Zeit. AWS Glue ETL-Aufträge stellen mehrere Features bereit, die Sie im ETL-Skript verwenden können, um das Schema und die Partitionen im Data Catalog zu aktualisieren. Mit diesen Features können Sie die Ergebnisse der ETL-Arbeit im Data Catalog anzeigen, ohne den Crawler erneut ausführen zu müssen.

Neue Partitionen

Wenn Sie die neuen Partitionen in anzeigen möchten AWS Glue Data Catalog, können Sie einen der folgenden Schritte ausführen:

- Führen Sie nach Abschluss des Auftrags den Crawler erneut aus und zeigen Sie die neuen Partitionen auf der Konsole an, wenn der Crawler beendet ist.
- Wenn der Auftrag abgeschlossen ist, können Sie sofort die neuen Partitionen auf der Konsole anzeigen, ohne den Crawler erneut ausführen zu müssen. Sie können dieses Feature aktivieren, indem Sie Ihrem ETL-Skript einige Codezeilen hinzufügen, wie in den folgenden Beispielen gezeigt. Der Code verwendet das `enableUpdateCatalog`-Argument, um anzuzeigen, dass der

Data Catalog während der Auftragsausführung aktualisiert werden soll, wenn die neuen Partitionen erstellt werden.

Methode 1

Übergeben Sie `enableUpdateCatalog` und `partitionKeys` an ein Optionsargument.

Python

```
additionalOptions = {"enableUpdateCatalog": True}
additionalOptions["partitionKeys"] = ["region", "year", "month", "day"]


sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<target_db_name>,

 table_name=<target_table_name>, transformation_ctx="write_sink",

 additional_options=additionalOptions)
```

Scala

```
val options = JsonOptions(Map(
    "path" -> <S3_output_path>,
    "partitionKeys" -> Seq("region", "year", "month", "day"),
    "enableUpdateCatalog" -> true))
val sink = glueContext.getCatalogSink(
    database = <target_db_name>,
    tableName = <target_table_name>,
    additionalOptions = options)sink.writeDynamicFrame(df)
```

Methode 2

Übergeben Sie `enableUpdateCatalog` und `partitionKeys` in `getSink()` und rufen Sie `setCatalogInfo()` auf dem `DataSink`-Objekt auf.

Python

```
sink = glueContext.getSink(
    connection_type="s3",
    path="<S3_output_path>",
    enableUpdateCatalog=True,
    partitionKeys=["region", "year", "month", "day"])
```

```
sink.setFormat("json")
sink.setCatalogInfo(catalogDatabase=<target_db_name>,
 catalogTableName=<target_table_name>)
sink.writeFrame(last_transform)
```

Scala

```
val options = JsonOptions(
   Map("path" -> <S3_output_path>,
       "partitionKeys" -> Seq("region", "year", "month", "day"),
       "enableUpdateCatalog" -> true))
val sink = glueContext.getSink("s3", options).withFormat("json")
sink.setCatalogInfo(<target_db_name>, <target_table_name>)
sink.writeDynamicFrame(df)
```

Jetzt können Sie neue Katalogtabellen erstellen, vorhandene Tabellen mit einem geänderten Schema aktualisieren und neue Tabellenpartitionen im Datenkatalog hinzufügen, indem Sie AWS Glue Der ETL-Job selbst, ohne dass Crawler erneut ausgeführt werden müssen.

Aktualisieren des Tabellenschemas

Wenn Sie das Schema der Data-Catalog-Tabelle überschreiben möchten, ist dies folgendermaßen möglich:

- Führen Sie nach Abschluss des Auftrags den Crawler erneut aus und stellen Sie sicher, dass der Crawler so konfiguriert ist, dass auch die Tabellendefinition aktualisiert wird. Zeigen Sie die neuen Partitionen in der Konsole zusammen mit allen Schemaaktualisierungen an, nachdem der Crawler beendet wurde. Weitere Informationen finden Sie unter Konfigurieren eines Crawlers mithilfe der API.
- Wenn der Auftrag endet, können Sie das geänderte Schema sofort in der Konsole anzeigen, ohne den Crawler erneut ausführen zu müssen. Sie können dieses Feature aktivieren, indem Sie Ihrem ETL-Skript einige Codezeilen hinzufügen, wie in den folgenden Beispielen gezeigt. Der Code verwendet `enableUpdateCatalog` mit dem Wert „true“ und `updateBehavior` mit dem Wert `UPDATE_IN_DATABASE`, damit während der Auftragsausführung das Schema überschrieben wird und neue Partitionen in den Data Catalog eingefügt werden.

Python

```
additionalOptions = {
```

```
    "enableUpdateCatalog": True,
    "updateBehavior": "UPDATE_IN_DATABASE"}
additionalOptions["partitionKeys"] = ["partition_key0", "partition_key1"]

sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<dst_db_name>,
    table_name=<dst_tbl_name>, transformation_ctx="write_sink",
    additional_options=additionalOptions)
job.commit()
```

Scala

```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("partition_0", "partition_1"),
    "enableUpdateCatalog" -> true))
val sink = glueContext.getCatalogSink(database = nameSpace, tableName = tableName,
 additionalOptions = options)
sink.writeDynamicFrame(df)
```

Sie können den `updateBehavior`-Wert auch auf `LOG` festlegen, wenn das Tabellenschema nicht überschrieben werden soll, die neuen Partitionen aber hinzugefügt werden sollen. Der Standardwert von `updateBehavior` ist `UPDATE_IN_DATABASE`. Wenn Sie also nicht explizit einen anderen Wert angeben, wird das Tabellenschema überschrieben.

Wenn `enableUpdateCatalog` nicht auf „true“ festgelegt ist, aktualisiert der ETL-Auftrag die Tabelle im Data Catalog nicht, unabhängig von der Option, die für `updateBehavior` ausgewählt ist.

Erstellen neuer Tabellen

Sie können dieselben Optionen auch verwenden, um eine neue Tabelle im Data Catalog zu erstellen. Sie können die Datenbank und den neuen Tabellennamen mit `setCatalogInfo` angeben.

Python

```
sink = glueContext.getSink(connection_type="s3", path="s3://path/to/data",
    enableUpdateCatalog=True, updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=["partition_key0", "partition_key1"])
sink.setFormat("<format>")
sink.setCatalogInfo(catalogDatabase=<dst_db_name>, catalogTableName=<dst_tbl_name>)
sink.writeFrame(last_transform)
```

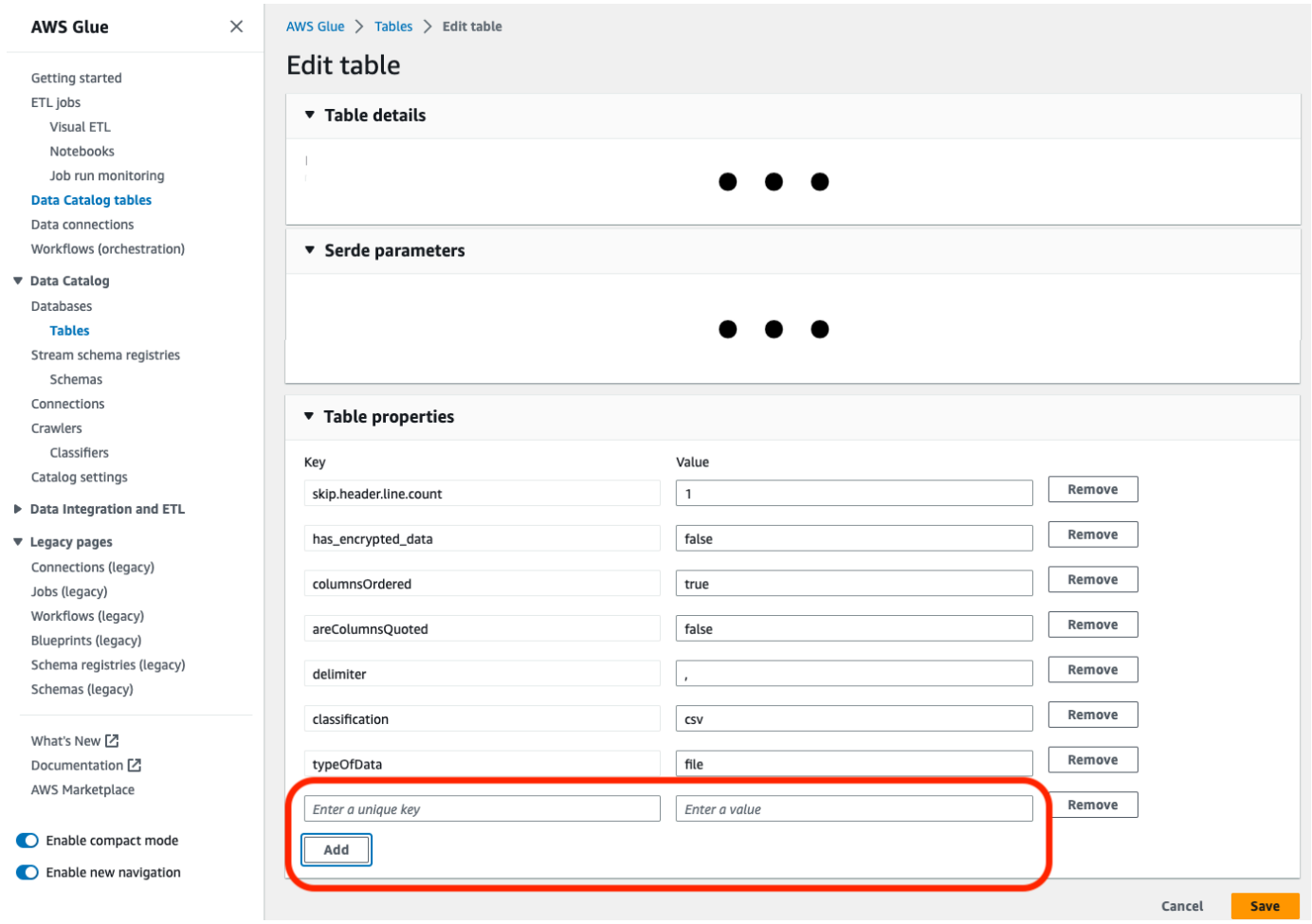
Scala

```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("<partition_1>", "<partition_2>"),
    "enableUpdateCatalog" -> true,
    "updateBehavior" -> "UPDATE_IN_DATABASE"))
val sink = glueContext.getSink(connectionType = "s3", connectionOptions =
 options).withFormat("<format>")
sink.setCatalogInfo(catalogDatabase = “<dst_db_name>”, catalogTableName =
 “<dst_tbl_name>”)
sink.writeDynamicFrame(df)
```

Einschränkungen

Beachten Sie die folgenden Einschränkungen:

- Es werden nur Amazon Simple Storage Service (Amazon S3)-Ziele unterstützt.
- Das `enableUpdateCatalog`-Feature wird für reglementierte Tabellen nicht unterstützt.
- Es werden nur die folgenden Formate unterstützt: `json`, `csv`, `avro` und `parquet`.
- Um Tabellen mit der `parquet` Klassifizierung zu erstellen oder zu aktualisieren, müssen Sie die AWS Glue optimierter Parkettschreiber für DynamicFrames. Dies kann mit einem der folgenden Schritte erreicht werden:
  - Wenn Sie eine vorhandene Tabelle im Katalog mit `parquet`-Klassifizierung aktualisieren, muss die `"useGlueParquetWriter"`-Tabelleneigenschaft der Tabelle auf `true` festgelegt sein, bevor Sie sie aktualisieren. Sie können diese Eigenschaft über das AWS Glue APIs /SDK, über die Konsole oder über eine Athena-DDL-Anweisung festlegen.



Sobald die Katalogtabelleneigenschaft festgelegt ist, können Sie den folgenden Codeausschnitt verwenden, um die Katalogtabelle mit den neuen Daten zu aktualisieren:

```
glueContext.write_dynamic_frame.from_catalog(
    frame=frameToWrite,
    database="dbName",
    table_name="tableName",
    additional_options={
        "enableUpdateCatalog": True,
        "updateBehavior": "UPDATE_IN_DATABASE"
    }
)
```

- Wenn die Tabelle noch nicht im Katalog vorhanden ist, können Sie die `getSink()`-Methode in Ihrem Skript mit `connection_type="s3"` verwenden, um die Tabelle und ihre Partitionen dem Katalog hinzuzufügen und die Daten in Amazon S3 zu schreiben. Geben Sie das entsprechende `partitionKeys` und `compression` für Ihren Workflow an.

```
s3sink = glueContext.getSink(
    path="s3://bucket/folder/",
    connection_type="s3",
    updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=[],
    compression="snappy",
    enableUpdateCatalog=True
)

s3sink.setCatalogInfo(
    catalogDatabase="dbName", catalogTableName="tableName"
)

s3sink.setFormat("parquet", useGlueParquetWriter=True)
s3sink.writeFrame(frameToWrite)
```

  - Der `glueparquet` Formatwert ist eine ältere Methode zur Aktivierung des AWS Glue Parquet Writers.
- Wenn `updateBehavior` auf `LOG` festgelegt ist, werden neue Partitionen nur hinzugefügt, wenn das `DynamicFrame`-Schema äquivalent zu den in der Data-Catalog-Tabelle definierten Spalten ist oder eine Teilmenge dieser Spalten enthält.
- Schemaaktualisierungen werden für nicht partitionierte Tabellen nicht unterstützt (ohne Verwendung der Option „partitionKeys“).
- Die partitionKeys müssen für den im ETL-Skript übergebenen Parameter und die partitionKeys im Tabellenschema des Data Catalogs äquivalent sein und in der gleichen Reihenfolge vorliegen.
- Dieses Feature unterstützt derzeit noch nicht das Aktualisieren/Erstellen von Tabellen, in denen die Aktualisierungsschemas verschachtelt sind (z. B. Arrays innerhalb von Strukturen).

Weitere Informationen finden Sie unter the section called “AWS Glue für Spark”.

Arbeiten mit MongoDB-Verbindungen in ETL-Aufträgen

Sie können eine Verbindung für MongoDB erstellen und diese Verbindung dann in Ihrem AWS Glue Beruf. Weitere Informationen finden Sie the section called “MongoDB-Verbindungen” im AWS Glue Programmierhandbuch. `url`, `username` und `password` für die Verbindung werden in der MongoDB-Verbindung gespeichert. Andere Optionen können in Ihrem ETL-Auftraggskript mit dem `additionalOptions`-Parameter von `glueContext.getCatalogSource` festgelegt werden. Weitere Optionen können Folgendes umfassen:

- `database`: (Erforderlich) Die MongoDB-Datenbank, aus der gelesen werden soll.
- `collection`: (Erforderlich) Die MongoDB-Sammlung, aus der gelesen werden soll.

Indem Sie die `database`- und `collection`-Informationen innerhalb des ETL-Auftragsskripts verwenden, können Sie dieselbe Verbindung für mehrere Aufträge verwenden.

1. Erstellen Sie eine AWS Glue Data Catalog Verbindung für die MongoDB-Datenquelle. Unter „connectionType“: „mongodb“ finden Sie eine Beschreibung der Verbindungsparameter. Sie können die Verbindung über die Konsole APIs oder CLI herstellen.
2. Erstellen Sie eine Datenbank in AWS Glue Data Catalog , um die Tabellendefinitionen für Ihre MongoDB-Daten zu speichern. Weitere Informationen finden Sie unter Erstellen von Datenbanken.
3. Erstellen Sie einen Crawler, der das Crawling der Daten in der MongoDB mit den Informationen in der Verbindung ausführt, um eine Verbindung mit der MongoDB herzustellen. Der Crawler erstellt die Tabellen in der AWS Glue Data Catalog , die die Tabellen in der MongoDB-Datenbank beschreiben, die Sie in Ihrem Job verwenden. Weitere Informationen finden Sie unter Verwenden von Crawlern zum Auffüllen des Datenkatalogs .
4. Erstellen Sie einen Auftrag mit einem benutzerdefinierten Skript. Sie können den Job mit der Konsole APIs oder der CLI erstellen. Weitere Informationen finden Sie unter Jobs hinzufügen in AWS Glue.
5. Wählen Sie die Datenziele Ihres Auftrags aus. Die Tabellen, die das Datenziel repräsentieren, können in Ihrem Data Catalog definiert werden, oder Ihr Auftrag kann die Zieltabellen erstellen, wenn er ausgeführt wird. Beim Verfassen des Auftrags wählen Sie einen Zielspeicherort aus. Wenn das Ziel eine Verbindung benötigt, wird auch die Verbindung in Ihrem Auftrag referenziert. Wenn Ihr Auftrag mehrere Datenziele benötigt, können Sie sie später hinzufügen, indem Sie das Skript bearbeiten.
6. Passen Sie die Auftragsverarbeitungsumgebung an, indem Sie Argumente für Ihren Auftrag und das generierte Skript bereitstellen.

   Hier sehen Sie ein Beispiel für das Erstellen eines `DynamicFrame` aus der MongoDB-Datenbank basierend auf der in Data Catalog definierten Tabellenstruktur. Der Code verwendet `additionalOptions` zur Bereitstellung der zusätzlichen Datenquelleninformationen:

   Scala

```
val resultFrame: DynamicFrame = glueContext.getCatalogSource(
```

```
        database = catalogDB,
        tableName = catalogTable,
        additionalOptions = JsonOptions(Map("database" -> DATABASE_NAME,
                "collection" -> COLLECTION_NAME))
      ).getDynamicFrame()
```

Python

```
glue_context.create_dynamic_frame_from_catalog(
        database = catalogDB,
        table_name = catalogTable,
        additional_options = {"database":"database_name",
            "collection":"collection_name"})
```

7. Führen Sie den Auftrag entweder on demand oder über einen Auslöser aus.

Optimieren der Abfrageleistung mithilfe von Spaltenstatistiken

Sie können Statistiken auf Spaltenebene für AWS Glue Data Catalog Tabellen in Datenformaten wie Parquet, ORC, JSON, ION, CSV und XML berechnen, ohne zusätzliche Datenpipelines einrichten zu müssen. Spaltenstatistiken helfen Ihnen dabei, Datenprofile zu verstehen, indem sie Einblicke in die Werte innerhalb einer Spalte gewinnen.

Der Datenkatalog unterstützt die Generierung von Statistiken für Spaltenwerte wie Minimalwert, Maximalwert, Gesamtzahl der Nullwerte, Gesamtzahl unterschiedlicher Werte, durchschnittliche Länge von Werten und Gesamtzahl der Vorkommen von wahren Werten. AWS Analysedienste wie Amazon Redshift und Amazon Athena können diese Spaltenstatistiken verwenden, um Abfrageausführungspläne zu generieren und den optimalen Plan auszuwählen, der die Abfrageleistung verbessert.

Es gibt drei Szenarien für die Generierung von Spaltenstatistiken:

Automatisch

AWS Glue unterstützt die automatische Generierung von Spaltenstatistiken auf Katalogebene, sodass automatisch Statistiken für neue Tabellen in der generiert werden können. AWS Glue Data Catalog

Geplant

AWS Glue unterstützt die Planung der Generierung von Spaltenstatistiken, sodass sie automatisch nach einem wiederkehrenden Zeitplan ausgeführt werden kann.

Bei der geplanten Statistikberechnung aktualisiert der Task für Spaltenstatistiken die Gesamtstatistiken auf Tabellenebene, z. B. Min., Max und Durchschnitt, mit den neuen Statistiken, sodass Abfrage-Engines genaue Statistiken und up-to-date Statistiken zur Optimierung der Abfrageausführung erhalten.

Auf Abruf

Verwenden Sie diese Option, um bei Bedarf Spaltenstatistiken auf Abruf zu generieren. Dies ist nützlich für Ad-hoc-Analysen oder wenn Statistiken sofort berechnet werden müssen.

Sie können so konfigurieren, dass die Aufgabe zur Generierung von Spaltenstatistiken mithilfe von AWS Glue Konsolen- und AWS CLI AWS Glue API-Operationen ausgeführt wird. Wenn Sie den Prozess starten, AWS Glue wird ein Spark-Job im Hintergrund gestartet und die AWS Glue Tabellenmetadaten im Datenkatalog aktualisiert. Sie können Spaltenstatistiken über die AWS Glue Konsole AWS CLI oder durch Aufrufen des GetColumnStatisticsForTableAPI-Vorgangs anzeigen.

> **Note**
>
> Wenn Sie Lake-Formation-Berechtigungen verwenden, um den Zugriff auf die Tabelle zu steuern, erfordert die von der Aufgabe für Spaltenstatistiken übernommene Rolle vollständigen Tabellenzugriff, um Statistiken zu generieren.

Themen

- Voraussetzungen für die Generierung von Spaltenstatistiken
- Automatische Generierung von Spaltenstatistiken
- Generieren von Spaltenstatistiken nach einem Zeitplan
- Generierung von Spaltenstatistiken auf Abruf
- Anzeigen von Spaltenstatistiken
- Anzeigen von Spaltenstatistik-Aufgabenläufen
- Beenden des Spaltenstatistik-Aufgabenlaufs
- Löschen von Spaltenstatistiken

- Überlegungen und Einschränkungen

Voraussetzungen für die Generierung von Spaltenstatistiken

Um Spaltenstatistiken zu erstellen oder zu aktualisieren, übernimmt die Statistikgenerierungsaufgabe in Ihrem Namen eine AWS Identity and Access Management (IAM)-Rolle. Basierend auf den der Rolle erteilten Berechtigungen kann die Aufgabe zur Generierung von Spaltenstatistiken die Daten aus dem Amazon-S3-Datenspeicher lesen.

Wenn Sie die Aufgabe zur Generierung von Spaltenstatistiken konfigurieren, AWS Glue können Sie eine Rolle erstellen, die die `AWSGlueServiceRole` AWS verwaltete Richtlinie sowie die erforderliche Inline-Richtlinie für die angegebene Datenquelle umfasst.

Wenn Sie eine bestehende Rolle für die Generierung von Spaltenstatistiken angeben, stellen Sie sicher, dass sie die `AWSGlueServiceRole` Richtlinie oder eine gleichwertige (oder eine eingeschränkte Version dieser Richtlinie) sowie die erforderlichen Inline-Richtlinien enthält. Gehen Sie wie folgt vor, um eine neue IAM-Rolle zu erstellen:

> **Note**
>
> Um Statistiken für von Lake Formation verwaltete Tabellen zu generieren, benötigt die IAM-Rolle, die zum Generieren von Statistiken verwendet wird, vollständigen Tabellenzugriff.

Wenn Sie die Aufgabe zur Generierung von Spaltenstatistiken konfigurieren, AWS Glue können Sie eine Rolle erstellen, die die `AWSGlueServiceRole` AWS verwaltete Richtlinie sowie die erforderliche Inline-Richtlinie für die angegebene Datenquelle umfasst. Sie können auch eine Rolle erstellen und die in der Richtlinie unten aufgeführten Berechtigungen anhängen und diese Rolle der Aufgabe zur Generierung von Spaltenstatistiken hinzufügen.

So erstellen Sie eine IAM-Rolle für die Generierung von Spaltenstatistiken

1. Eine Anleitung zum Erstellen einer IAM;-Rolle finden Sie unter Erstellen von IAM-Rollen für AWS Glue.
2. Um eine bestehende Rolle zu aktualisieren, wechseln Sie in der IAM-Konsole zu der IAM-Rolle, die beim Generieren von Spaltenstatistiken verwendet wird.

3. Wählen Sie in der Registerkarte Berechtigungen hinzufügen die Option Richtlinien anfügen aus. Wählen Sie im neu geöffneten Browserfenster die Option `AWSGlueServiceRole` AWS Verwaltete Richtlinie aus.
4. Sie müssen außerdem Berechtigungen zum Lesen von Daten am Amazon-S3-Speicherort hinzufügen.

   Wählen Sie im Abschnitt Berechtigungen hinzufügen die Option Richtlinie erstellen aus. Erstellen Sie im neu geöffneten Browserfenster eine neue Richtlinie, die Sie mit Ihrer Rolle verwenden möchten.
5. Wählen Sie auf der Seite Richtlinie erstellen die Registerkarte JSON aus. Kopieren Sie den folgenden `JSON`-Code in das Richtlinien-Editor-Feld.

   > **Note**
   >
   > Ersetzen Sie in den folgenden Richtlinien die Konto-ID durch eine gültige AWS-Konto und `region` durch die Region der Tabelle sowie `bucket-name` durch den Amazon S3 S3-Bucket-Namen.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3BucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>/*",
                "arn:aws:s3:::<bucket-name>"
            ]
        }
     ]
}
```

6. (Optional) Wenn Sie Lake-Formation-Berechtigungen verwenden, um Zugriff auf Ihre Daten zu gewähren, sind für die IAM-Rolle `lakeformation:GetDataAccess`-Berechtigungen erforderlich.

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
          {
               "Sid": "LakeFormationDataAccess",
               "Effect": "Allow",
               "Action": "lakeformation:GetDataAccess",
               "Resource": [
                   "*"
               ]
           }
       ]
   }
   ```

   Wenn der Amazon-S3-Speicherort bei Lake Formation registriert ist und für die IAM-Rolle, die von der Aufgabe zur Generierung von Spaltenstatistiken übernommen wurde, keine `IAM_ALLOWED_PRINCIPALS`-Gruppenberechtigungen für die Tabelle erteilt wurden, benötigt die Rolle `ALTER`- und `DESCRIBE`-Berechtigungen in Lake Formation für die Tabelle. Die Rolle, die für die Registrierung des Amazon-S3-Buckets verwendet wird, erfordert `INSERT`- und `DELETE`-Berechtigungen in Lake Formation für die Tabelle.

   Wenn der Amazon-S3-Speicherort bei Lake Formation registriert ist und für die IAM-Rolle keine `IAM_ALLOWED_PRINCIPALS`-Gruppenberechtigungen für die Tabelle erteilt wurden, benötigt die Rolle `ALTER`-, `DESCRIBE`-, `INSERT`- und `DELETE`-Berechtigungen in Lake Formation für die Tabelle.

7. Wenn Sie die `Automatic statistics generation` Option auf Katalogebene aktiviert haben, muss die IAM-Rolle über die `glue:UpdateCatalog` Berechtigung oder die Lake Formation `ALTER CATALOG` Formation-Berechtigung für den Standarddatenkatalog verfügen. Sie können den `GetCatalog` Vorgang verwenden, um die Katalogeigenschaften zu überprüfen.

8. (Optional) Für die Aufgabe zur Generierung von Spaltenstatistiken, die verschlüsselte Amazon CloudWatch Logs -Schreibvorgänge durchführt, sind die folgenden Berechtigungen in der Schlüsselrichtlinie erforderlich.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "CWLogsKmsPermissions",
        "Effect":  "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents",
            "logs:AssociateKmsKey"
        ],
        "Resource": [
            "arn:aws:logs:<region>:111122223333:log-group:/aws-glue:*"
        ]
    },
    {
        "Sid": "KmsPermissions",
        "Effect": "Allow",
        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt",
            "kms:Encrypt"
        ],
        "Resource": [
        "arn:aws:kms:<region>:111122223333:key/"arn of key used for ETL cloudwatch
 encryption"
        ],
        "Condition": {
            "StringEquals": {
                "kms:ViaService": ["glue.<region>.amazonaws.com"]
            }
        }
    }
    ]
}
```

9. Die Rolle, die Sie zum Ausführen von Spaltenstatistiken verwenden, muss über die `iam:PassRole` entsprechende Berechtigung verfügen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:PassRole"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:role/<columnstats-role-name>"
        ]
    }]
}
```

10. Wenn Sie eine IAM-Rolle für die Generierung von Spaltenstatistiken erstellen, muss für diese Rolle auch die folgende Vertrauensrichtlinie gelten, damit der Service die Rolle übernehmen kann.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Sid": "TrustPolicy",
        "Effect": "Allow",
        "Principal": {
          "Service": "glue.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
      }
   ]
}
```

Automatische Generierung von Spaltenstatistiken

Die automatische Generierung von Spaltenstatistiken ermöglicht es Ihnen, Statistiken für neue Tabellen in der zu planen und automatisch zu berechnen AWS Glue Data Catalog. Wenn Sie die automatische Statistikgenerierung aktivieren, erkennt der Datenkatalog neue Tabellen mit

bestimmten Datenformaten wie Parquet, JSON, CSV, XML, ORC, ION und Apache Iceberg zusammen mit ihren individuellen Bucket-Pfaden. Bei einer einmaligen Katalogkonfiguration generiert der Datenkatalog Statistiken für diese Tabellen.

Data Lake-Administratoren können die Statistikgenerierung konfigurieren, indem sie den Standardkatalog in der Lake Formation Formation-Konsole auswählen und mithilfe der `Optimization configuration` Option Tabellenstatistiken aktivieren. Wenn Sie neue Tabellen erstellen oder bestehende Tabellen im Datenkatalog aktualisieren, erfasst der Datenkatalog wöchentlich die Anzahl der unterschiedlichen Werte (NDVs) für Apache Iceberg-Tabellen sowie zusätzliche Statistiken wie die Anzahl der Nullen, die maximale, minimale und durchschnittliche Länge für andere unterstützte Dateiformate.

Wenn Sie die Statistikgenerierung auf Tabellenebene konfiguriert haben oder wenn Sie zuvor die Einstellungen für die Statistikgenerierung für eine Tabelle gelöscht haben, haben diese tabellenspezifischen Einstellungen Vorrang vor den Standardkatalogeinstellungen für die automatische Generierung von Spaltenstatistiken.

Die Aufgabe zur automatischen Statistikgenerierung analysiert 50% der Datensätze in den Tabellen, um Statistiken zu berechnen. Die automatische Generierung von Spaltenstatistiken stellt sicher, dass der Datenkatalog wöchentliche Kennzahlen enthält, die von Abfrage-Engines wie Amazon Athena und Amazon Redshift Spectrum verwendet werden können, um die Abfrageleistung zu verbessern und potenzielle Kosteneinsparungen zu erzielen. Es ermöglicht die Planung der Statistikgenerierung mithilfe der Konsole AWS Glue APIs oder der Konsole und bietet so einen automatisierten Prozess ohne manuelles Eingreifen.

Themen

- Automatische Statistikgenerierung auf Katalogebene aktivieren
- Automatisierte Einstellungen auf Tabellenebene anzeigen
- Deaktivierung der Generierung von Spaltenstatistiken auf Katalogebene

Automatische Statistikgenerierung auf Katalogebene aktivieren

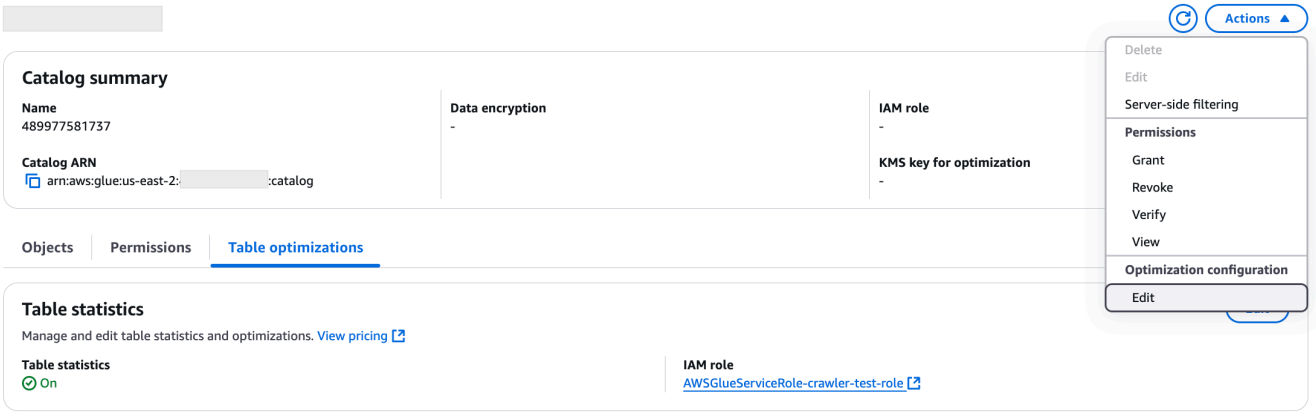
Sie können die automatische Generierung von Spaltenstatistiken für alle neuen Apache Iceberg-Tabellen und Tabellen in Nicht-OTF-Tabellenformaten (Parquet, JSON, CSV, XML, ORC, ION) im Datenkatalog aktivieren. Nach dem Erstellen der Tabelle können Sie die Einstellungen für die Spaltenstatistik auch explizit manuell aktualisieren.

Um die Datenkatalogeinstellungen so zu aktualisieren, dass sie die Katalogebene aktivieren, muss die verwendete IAM-Rolle über die `glue:UpdateCatalog` entsprechende Berechtigung oder AWS Lake Formation `ALTER CATALOG` Berechtigung für den Stammkatalog verfügen. Sie können die `GetCatalog` API verwenden, um die Katalogeigenschaften zu überprüfen.

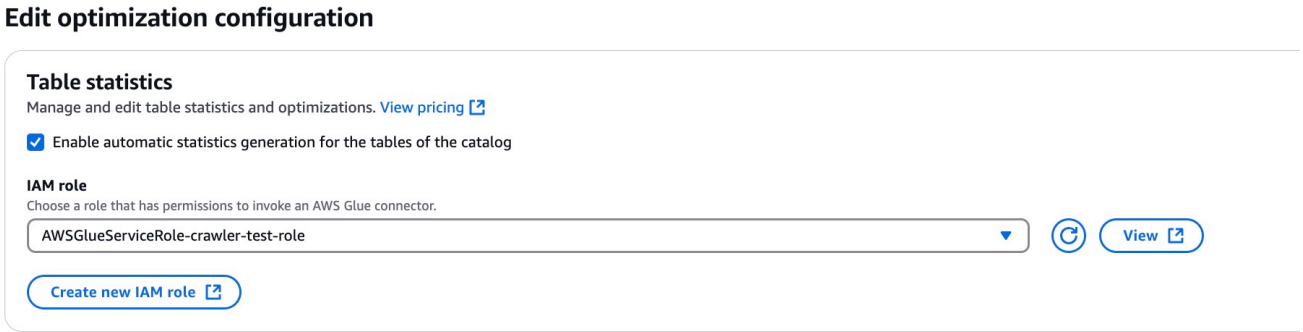
AWS Management Console

Um die automatische Generierung von Spaltenstatistiken auf Kontoebene zu aktivieren

1. Öffnen Sie die Lake Formation Formation-Konsole unter https://console.aws.amazon.com/lakeformation/.
2. Wählen Sie in der linken Navigationsleiste Kataloge aus.
3. Wählen Sie auf der Seite mit der Katalogübersicht unter Optimierungskonfiguration die Option Bearbeiten aus.



4. Wählen Sie auf der Konfigurationsseite für die Tabellenoptimierung die Option Automatische Statistikgenerierung für die Tabellen des Katalogs aktivieren aus.



5. Wählen Sie eine vorhandene IAM-Rolle aus, oder erstellen Sie eine neue, die über die erforderlichen Berechtigungen verfügt, um die Aufgabe für Spaltenstatistiken auszuführen.

6. Wählen Sie Absenden aus.

AWS CLI

Sie können die Erfassung von Statistiken auf Katalogebene auch über die aktivieren. AWS CLI Führen Sie den folgenden Befehl aus, um die Statistikerfassung auf Tabellenebene mit AWS CLI zu konfigurieren:

```
aws glue update-catalog --cli-input-json '{
    "name": "123456789012",
    "catalogInput": {
        "description": "Updating root catalog with role arn",
        "catalogProperties": {
            "customProperties": {
                "ColumnStatistics.RoleArn": "arn:aws:iam::"123456789012":role/
service-role/AWSGlueServiceRole",
                "ColumnStatistics.Enabled": "true"
            }
        }
    }
}'
```

Der obige Befehl ruft AWS Glue die `UpdateCatalog` Operation auf, die eine `CatalogProperties` Struktur mit den folgenden Schlüssel-Wert-Paaren für die Generierung von Statistiken auf Katalogebene verwendet:

- ColumnStatistics. RoleArn — ARN der IAM-Rolle, der für alle Aufgaben verwendet werden soll, die für die Generierung von Statistiken auf Katalogebene ausgelöst wurden
- ColumnStatistics.Enabled — Boolescher Wert, der angibt, ob die Einstellungen auf Katalogebene aktiviert oder deaktiviert sind

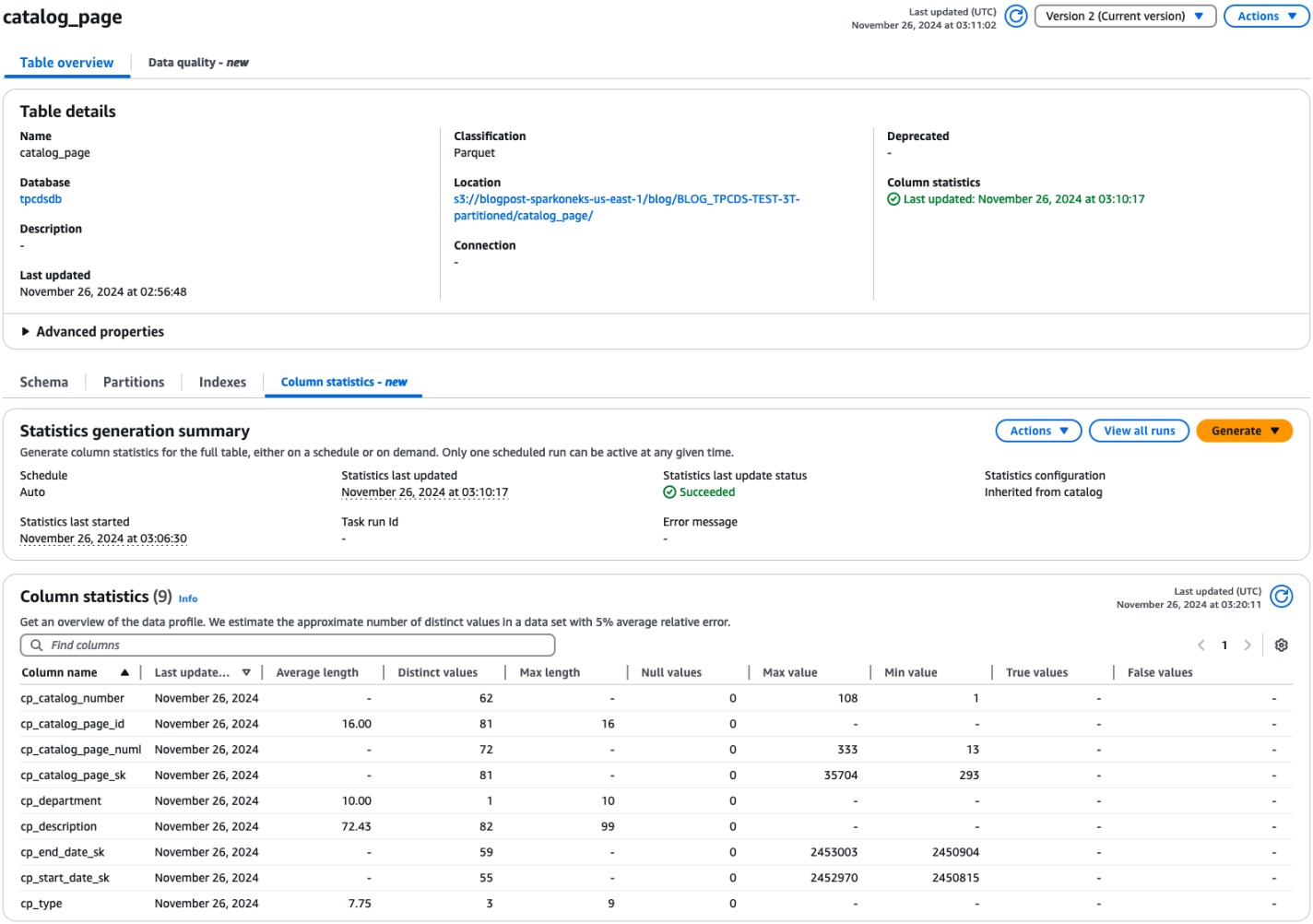
Automatisierte Einstellungen auf Tabellenebene anzeigen

Wenn die Erfassung von Statistiken auf Katalogebene aktiviert ist, wird bei jeder Erstellung oder Aktualisierung einer Apache Hive-Tabelle oder einer Apache Iceberg-Tabelle über das SDK `CreateTable` oder `UpdateTable` APIs über AWS Management Console das SDK oder AWS-Glue-Crawler eine entsprechende Einstellung auf Tabellenebene für diese Tabelle erstellt.

Tabellen mit aktivierter automatischer Statistikgenerierung müssen eine der folgenden Eigenschaften aufweisen:

- Verwenden Sie eine`InputSerdeLibrary`, die mit org.apache.hadoop beginnt und entspricht `TableType EXTERNAL_TABLE`
- Verwenden Sie eine, die mit und gleich beginnt `InputSerdeLibrary com.amazon.ion` `TableType EXTERNAL_TABLE`
- Enthält table_type: „ICEBERG“ in seiner Parameterstruktur.

Nachdem Sie eine Tabelle erstellt oder aktualisiert haben, können Sie die Tabellendetails überprüfen, um die Statistikgenerierung zu bestätigen. `Statistics generation summary`Zeigt die `Schedule` Eigenschaft an, die als festgelegt ist AUTO und `Statistics configuration` deren Wert lautet`Inherited from catalog`. Jede Tabelleneinstellung mit der folgenden Einstellung würde automatisch intern von Glue ausgelöst.



Deaktivierung der Generierung von Spaltenstatistiken auf Katalogebene

Sie können die automatische Generierung von Spaltenstatistiken für neue Tabellen mithilfe der AWS Lake Formation Konsole, der API oder der `glue:UpdateCatalogSettings` API deaktivieren. `glue:DeleteColumnStatisticsTaskSettings`

Um die automatische Generierung von Spaltenstatistiken auf Kontoebene zu deaktivieren

1. Öffnen Sie die Lake Formation Formation-Konsole unter https://console.aws.amazon.com/lakeformation/.
2. Wählen Sie in der linken Navigationsleiste Kataloge aus.
3. Wählen Sie auf der Seite mit der Katalogübersicht unter Optimierungskonfiguration die Option Bearbeiten aus.
4. Deaktivieren Sie auf der Konfigurationsseite für die Tabellenoptimierung die Option Automatische Statistikgenerierung für die Tabellen des Katalogs aktivieren.
5. Wählen Sie Absenden aus.

Generieren von Spaltenstatistiken nach einem Zeitplan

Gehen Sie wie folgt vor, um einen Zeitplan für die Generierung von Spaltenstatistiken AWS Glue Data Catalog mithilfe der AWS Glue Konsole AWS CLI, der oder der CreateColumnStatisticsTaskSettingsOperation zu konfigurieren.

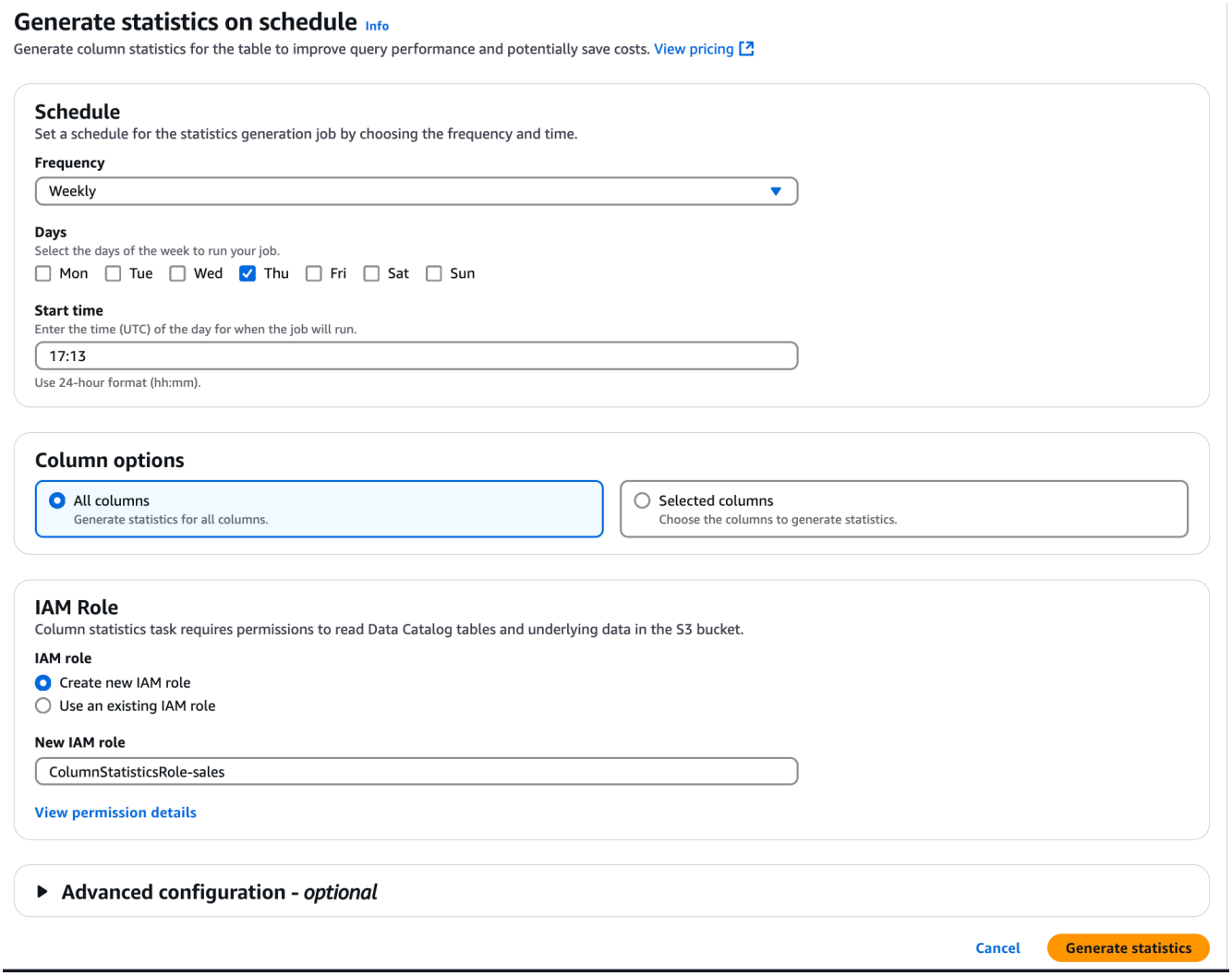
Console

Generieren von Spaltenstatistiken mit der Konsole

1. Melden Sie sich bei der AWS Glue Konsole unter an https://console.aws.amazon.com/glue/.
2. Wählen Sie „Datenkatalog-Tabellen“ aus.
3. Wählen Sie eine Tabelle aus der Liste aus.
4. Wählen Sie im unteren Bereich der Tabellenseite den Tab Spaltenstatistiken aus.
5. Sie können unter „Aktionen“ auch unter „Spaltenstatistiken“ die Option „Nach Zeitplan generieren“ auswählen.
6. Konfigurieren Sie auf der Seite Statistiken nach Zeitplan generieren einen wiederkehrenden Zeitplan für die Ausführung der Aufgabe Spaltenstatistik, indem Sie die Häufigkeit und die Startzeit auswählen. Sie können wählen, ob die Häufigkeit stündlich, täglich oder wöchentlich sein soll, oder Sie können einen Cron-Ausdruck definieren, um den Zeitplan anzugeben.

Ein Cron-Ausdruck ist eine Zeichenfolge, die ein Zeitplanmuster darstellt und aus 6 durch Leerzeichen getrennten Feldern besteht: * * * * * <minute><hour><day of month><month><day of week><year>Um beispielsweise eine Aufgabe jeden Tag um Mitternacht auszuführen, wäre der Cron-Ausdruck: 0 0 * *? *

Weitere Informationen finden Sie unter Cron-Ausdrücke.



7. Wählen Sie als Nächstes die Spaltenoption, um Statistiken zu generieren.

   - Alle Spalten — Wählen Sie diese Option, um Statistiken für alle Spalten in der Tabelle zu generieren.
   - Ausgewählte Spalten – Wählen Sie diese Option aus, um Statistiken für bestimmte Spalten zu generieren. Sie können die Spalten in der Dropdown-Liste auswählen.

8. Wählen Sie eine IAM-Rolle oder erstellen Sie eine bestehende Rolle, die über Berechtigungen zum Generieren von Statistiken verfügt. AWS Glue übernimmt diese Rolle, um Spaltenstatistiken zu generieren.

   Eine schnellere Methode besteht darin, die AWS Glue Konsole eine Rolle für Sie erstellen zu lassen. Die von ihr erstellte Rolle dient speziell der Generierung von Spaltenstatistiken und umfasst die `AWSGlueServiceRole` AWS verwaltete Richtlinie sowie die erforderliche Inline-Richtlinie für die angegebene Datenquelle.

   Wenn Sie eine bestehende Rolle für die Generierung von Spaltenstatistiken angeben, stellen Sie sicher, dass sie die `AWSGlueServiceRole` Richtlinie oder eine gleichwertige (oder eine Version mit eingeschränktem Umfang) sowie die erforderlichen Inline-Richtlinien enthält.
9. (Optional) Wählen Sie als Nächstes eine Sicherheitskonfiguration aus, um für Protokolle die Verschlüsselung im Ruhezustand zu aktivieren.
10. (Optional) Sie können eine Stichprobengröße wählen, indem Sie nur einen bestimmten Prozentsatz der Zeilen aus der Tabelle angeben, um Statistiken zu generieren. Der Standardwert ist „Alle Zeilen“. Verwenden Sie die Aufwärts- und Abwärtspfeile, um den Prozentwert zu erhöhen oder zu verringern.

    Wir empfehlen, alle Zeilen in die Tabelle aufzunehmen, um genaue Statistiken zu berechnen. Verwenden Sie Beispielzeilen zur Generierung von Spaltenstatistiken nur dann, wenn Näherungswerte akzeptabel sind.
11. Wählen Sie Statistik generieren aus, um die Aufgabe zur Generierung von Spaltenstatistiken auszuführen.

AWS CLI

Sie können das folgende AWS CLI Beispiel verwenden, um einen Zeitplan für die Generierung von Spaltenstatistiken zu erstellen. Datenbankname, Tabellenname und Rolle sind erforderliche Parameter, und optionale Parameter sind Zeitplan, Katalog-ID, Stichprobengröße und column-name-list Sicherheitskonfiguration.

```
aws glue create-column-statistics-task-settings \
 --database-name 'database_name' \
 --table-name table_name \
 --role 'arn:aws:iam::123456789012:role/stats-role' \
 --schedule 'cron(0 0-5 14 * * ?)' \
 --column-name-list 'col-1' \
```

```
--catalog-id '123456789012' \
--sample-size '10.0 ' \
--security-configuration 'test-security'
```

Sie können Spaltenstatistiken auch generieren, indem Sie den Vorgang aufrufen. StartColumnStatisticsTaskRun

Verwaltung des Zeitplans für die Generierung von Spaltenstatistiken

Sie können die Planungsvorgänge wie das Aktualisieren, Starten, Stoppen und Löschen von Zeitplänen für die Generierung von Spaltenstatistiken in verwalten AWS Glue. Sie können AWS Glue Konsolen AWS CLI- oder API-Operationen für AWS Glue Spaltenstatistiken verwenden, um diese Aufgaben auszuführen.

Themen

- Aktualisierung des Zeitplans für die Generierung von Spaltenstatistiken
- Den Zeitplan für die Generierung von Spaltenstatistiken beenden
- Fortsetzung des Zeitplans für die Generierung von Spaltenstatistiken
- Der Zeitplan für die Generierung von Spaltenstatistiken wird gelöscht

Aktualisierung des Zeitplans für die Generierung von Spaltenstatistiken

Sie können den Zeitplan aktualisieren, um die Aufgabe zur Generierung der Spaltenstatistiken auszulösen, nachdem er erstellt wurde. Sie können die AWS Glue Konsole verwenden oder den UpdateColumnStatisticsTaskSettingsVorgang ausführen AWS CLI, um den Zeitplan für eine Tabelle zu aktualisieren. Sie können die Parameter eines vorhandenen Zeitplans ändern, z. B. den Zeitplantyp (auf Anforderung oder geplant) und andere optionale Parameter.

AWS Management Console

Um die Einstellungen für eine Aufgabe zur Generierung von Spaltenstatistiken zu aktualisieren

1. Melden Sie sich bei der AWS Glue Konsole an unter https://console.aws.amazon.com/glue/.
2. Wählen Sie die Tabelle, die Sie aktualisieren möchten, aus der Tabellenliste aus.
3. Wählen Sie im unteren Bereich der Tabellendetailseite die Option Spaltenstatistik aus.
4. Wählen Sie unter Aktionen die Option Bearbeiten aus, um den Zeitplan zu aktualisieren.

5. Nehmen Sie die gewünschten Änderungen am Zeitplan vor und wählen Sie Speichern.

AWS CLI

Wenn Sie die Funktion zur Generierung AWS Glue von Statistiken in der Konsole nicht verwenden, können Sie den Zeitplan mithilfe des `update-column-statistics-task-settings` Befehls manuell aktualisieren. Im folgenden Beispiel wird gezeigt, wie Sie Spaltenstatistiken mithilfe von AWS CLI aktualisieren.

```
aws glue update-column-statistics-task-settings \
 --database-name 'database_name' \
 --table-name 'table_name' \
 --role arn:aws:iam::123456789012:role/stats_role \
 --schedule 'cron(0 0-5 16 * * ?)' \
 --column-name-list 'col-1' \
 --sample-size '20.0' \
 --catalog-id '123456789012'\
 --security-configuration 'test-security'
```

Den Zeitplan für die Generierung von Spaltenstatistiken beenden

Wenn Sie die inkrementellen Statistiken nicht mehr benötigen, können Sie die geplante Generierung beenden, um Ressourcen und Kosten zu sparen. Das Anhalten des Zeitplans hat keine Auswirkungen auf die zuvor generierten Statistiken. Sie können den Zeitplan nach Belieben fortsetzen.

AWS Management Console

Um den Zeitplan für eine Aufgabe zur Generierung von Spaltenstatistiken zu beenden

1. Wählen Sie in der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie eine Tabelle mit Spaltenstatistiken aus.
3. Wählen Sie auf der Seite Tabellendetails die Option Spaltenstatistiken aus.
4. Wählen Sie unter Aktionen die Option Geplante Generierung, Pause aus.
5. Wählen Sie zur Bestätigung Pause aus.

AWS CLI

Um die Ausführung einer Aufgabe mit Spaltenstatistiken mithilfe von zu beenden AWS CLI, können Sie den folgenden Befehl verwenden:

```
aws glue stop-column-statistics-task-run-schedule \
 --database-name ''database_name' \
 --table-name 'table_name'
```

Ersetzen Sie das `database_name` und das `table_name` durch die tatsächlichen Namen der Datenbank und Tabelle, für die Sie den Zeitplan für die Ausführung der Aufgabe für Spaltenstatistiken beenden möchten.

Fortsetzung des Zeitplans für die Generierung von Spaltenstatistiken

Wenn Sie den Zeitplan für die Statistikgenerierung angehalten haben, AWS Glue können Sie den Zeitplan nach Belieben fortsetzen. Sie können den Zeitplan über die AWS Glue Konsole oder den AWS CLIStartColumnStatisticsTaskRunScheduleVorgang fortsetzen.

AWS Management Console

Um den Zeitplan für die Generierung von Spaltenstatistiken fortzusetzen

1. Wählen Sie auf der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie eine Tabelle mit Spaltenstatistiken aus.
3. Wählen Sie auf der Seite Tabellendetails die Option Spaltenstatistiken aus.
4. Wählen Sie unter Aktionen die Option Geplante Generierung und anschließend Fortfahren aus.
5. Wählen Sie zur Bestätigung „Fortfahren“.

AWS CLI

Ersetzen Sie `database_name` und die `table_name` durch die tatsächlichen Namen der Datenbank und Tabelle, für die Sie den Zeitplan für die Ausführung der Aufgabe für Spaltenstatistiken beenden möchten.

```
aws glue start-column-statistics-task-run-schedule \
```

```
--database-name 'database_name' \
--table-name 'table_name'
```

Der Zeitplan für die Generierung von Spaltenstatistiken wird gelöscht

Für eine optimale Abfrageleistung wird zwar generell empfohlen, up-to-date Statistiken zu verwalten, es gibt jedoch spezielle Anwendungsfälle, in denen das Entfernen des Zeitplans für die automatische Generierung von Vorteil sein könnte.

- Wenn die Daten relativ statisch bleiben, bleiben die vorhandenen Spaltenstatistiken möglicherweise über einen längeren Zeitraum korrekt, sodass weniger häufige Aktualisierungen erforderlich sind. Durch das Löschen des Zeitplans können unnötiger Ressourcenverbrauch und Mehraufwand vermieden werden, der mit der Neugenerierung von Statistiken über unveränderliche Daten verbunden ist.
- Wenn die manuelle Steuerung der Statistikgenerierung bevorzugt wird. Durch das Löschen des automatischen Zeitplans können Administratoren die Spaltenstatistiken in bestimmten Intervallen oder nach erheblichen Datenänderungen selektiv aktualisieren und so den Prozess an ihre Wartungsstrategien und den Bedarf an der Ressourcenzuweisung ausrichten.

AWS Management Console

Um den Zeitplan für die Generierung von Spaltenstatistiken zu löschen

1. Wählen Sie auf der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie eine Tabelle mit Spaltenstatistiken aus.
3. Wählen Sie auf der Seite Tabellendetails die Option Spaltenstatistiken aus.
4. Wählen Sie unter Aktionen die Option Geplante Generierung, Löschen aus.
5. Wählen Sie zur Bestätigung Löschen.

AWS CLI

Ersetzen Sie `database_name` und die `table_name` durch die tatsächlichen Namen der Datenbank und Tabelle, für die Sie den Zeitplan für die Ausführung der Aufgabe für Spaltenstatistiken beenden möchten.

Sie können den Zeitplan für Spaltenstatistiken mithilfe der DeleteColumnStatisticsTaskSettingsAPI-Operation oder löschen AWS CLI. Das folgende Beispiel

zeigt, wie der Zeitplan für die Generierung von Spaltenstatistiken mithilfe von AWS Command Line Interface (AWS CLI) gelöscht wird.

```
aws glue delete-column-statistics-task-settings \
    --database-name 'database_name' \
    --table-name 'table_name'
```

Generierung von Spaltenstatistiken auf Abruf

Sie können die Aufgabe Spaltenstatistiken für die AWS Glue Data Catalog Tabellen-Aufgabe bei Bedarf ohne festen Zeitplan ausführen. Diese Option ist nützlich für Ad-hoc-Analysen oder wenn Statistiken sofort berechnet werden müssen.

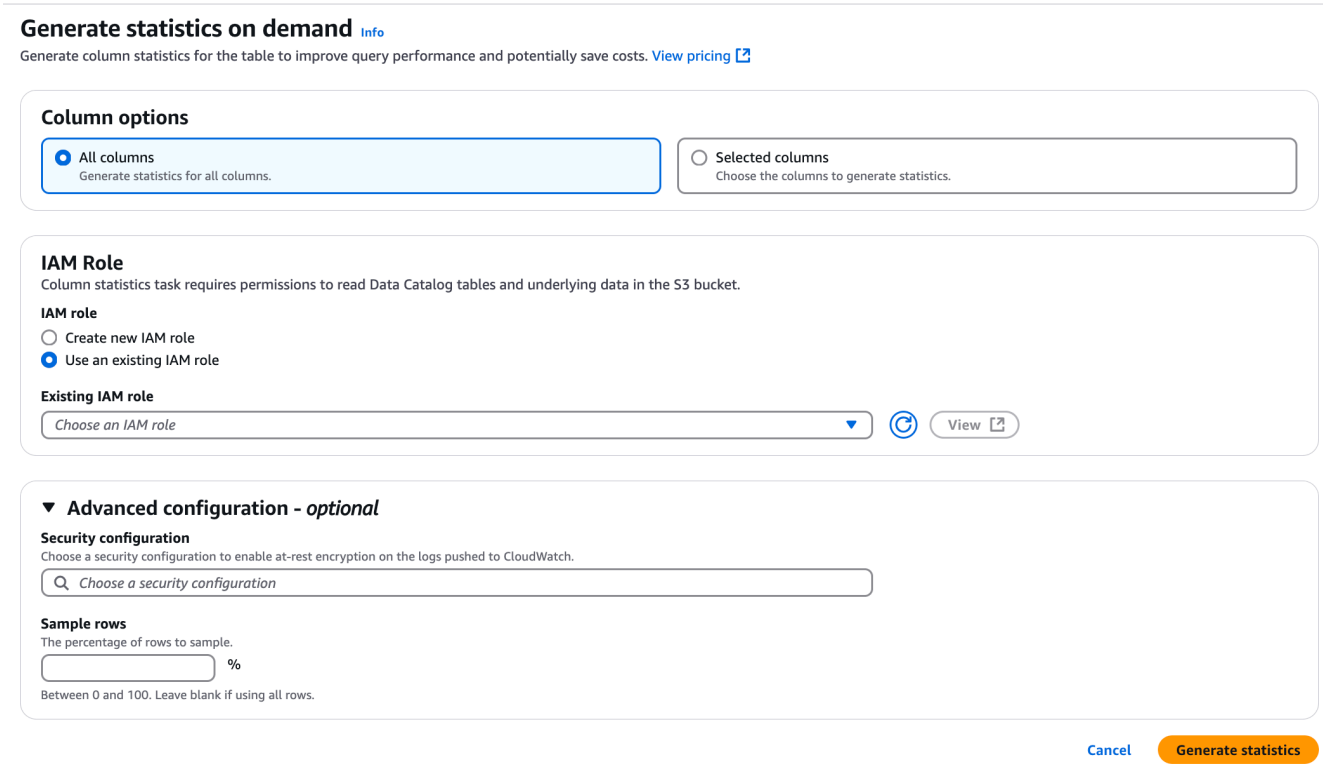
Gehen Sie wie folgt vor, um bei Bedarf Spaltenstatistiken für die Datenkatalogtabellen mithilfe der AWS Glue Konsole oder zu generieren. AWS CLI

AWS Management Console

Generieren von Spaltenstatistiken mit der Konsole

1. Melden Sie sich bei der AWS Glue Konsole an unter https://console.aws.amazon.com/glue/.
2. Wählen Sie „Datenkatalog-Tabellen“ aus.
3. Wählen Sie eine Tabelle aus der Liste aus.
4. Wählen Sie im Menü Aktionen die Option Statistiken generieren aus.

   Sie können auch auf der Registerkarte Spaltenstatistiken im unteren Bereich der Tabellenseite die Option Generieren, Generate on demand auswählen.
5. Folgen Sie den Schritten 7 bis 11 unterGenerieren von Spaltenstatistiken nach einem Zeitplan, um Spaltenstatistiken für die Tabelle zu generieren.
6. Geben Sie auf der Seite Statistiken generieren die folgenden Optionen an:



- Alle Spalten — Wählen Sie diese Option, um Statistiken für alle Spalten in der Tabelle zu generieren.
- Ausgewählte Spalten – Wählen Sie diese Option aus, um Statistiken für bestimmte Spalten zu generieren. Sie können die Spalten in der Dropdown-Liste auswählen.
- IAM-Rolle — Wählen Sie Neue IAM-Rolle erstellen, die über die erforderlichen Berechtigungsrichtlinien verfügt, um die Aufgabe zur Generierung von Spaltenstatistiken auszuführen. Wählen Sie „Berechtigungsdetails anzeigen“, um die Richtlinienerklärung zu überprüfen. Sie können auch eine IAM-Rolle aus der Liste auswählen. Weitere Informationen zu den erforderlichen Berechtigungen finden Sie unter Voraussetzungen für die Generierung von Spaltenstatistiken.

  AWS Glue nimmt die Berechtigungen der Rolle an, die Sie zum Generieren von Statistiken angeben.

  Weitere Informationen zur Bereitstellung von Rollen für AWS Glue finden Sie unter Identitätsbasierte Richtlinien für. AWS Glue.
- (Optional) Wählen Sie als Nächstes eine Sicherheitskonfiguration aus, um für Protokolle die Verschlüsselung im Ruhezustand zu aktivieren.

- Beispielzeilen – Wählen Sie nur einen bestimmten Prozentsatz der Zeilen aus der Tabelle aus, um Statistiken zu generieren. Der Standardwert ist „Alle Zeilen“. Verwenden Sie die Aufwärts- und Abwärtspfeile, um den Prozentwert zu erhöhen oder zu verringern.

  > **Note**
  >
  > Wir empfehlen, alle Zeilen in die Tabelle aufzunehmen, um genaue Statistiken zu berechnen. Verwenden Sie Beispielzeilen zur Generierung von Spaltenstatistiken nur dann, wenn Näherungswerte akzeptabel sind.

Wählen Sie Statistik generieren, um die Aufgabe auszuführen.

AWS CLI

Dieser Befehl löst die Ausführung einer Aufgabe mit Spaltenstatistiken für die angegebene Tabelle aus. Sie müssen den Datenbanknamen, den Tabellennamen, eine IAM-Rolle mit Berechtigungen zum Generieren von Statistiken und optional Spaltennamen und einen Prozentsatz der Stichprobengröße für die Statistikberechnung angeben.

```
aws glue start-column-statistics-task-run \
    --database-name 'database_name \
    --table-name 'table_name' \
    --role 'arn:aws:iam::123456789012:role/stats-role' \
    --column-name 'col1','col2'  \
    --sample-size 10.0
```

Dieser Befehl startet eine Aufgabe zum Generieren von Spaltenstatistiken für die angegebene Tabelle.

Aktualisierung der Spaltenstatistiken bei Bedarf

Die Pflege von up-to-date Spaltenstatistiken ist für den Abfrageoptimierer von entscheidender Bedeutung, um effiziente Ausführungspläne zu erstellen und so eine verbesserte Abfrageleistung, einen geringeren Ressourcenverbrauch und eine insgesamt bessere Systemleistung sicherzustellen. Dieser Prozess ist besonders wichtig nach erheblichen Datenänderungen, wie z. B. Massenladungen oder umfangreichen Änderungen, die bestehende Statistiken überflüssig machen können.

Sie müssen die Aufgabe Statistik generieren explizit von der AWS Glue Konsole aus ausführen, um die Spaltenstatistiken zu aktualisieren. Der Datenkatalog aktualisiert die Statistiken nicht automatisch.

Wenn Sie die Funktion zur Generierung AWS Glue von Statistiken in der Konsole nicht verwenden, können Sie die Spaltenstatistiken mithilfe der UpdateColumnStatisticsForTableAPI-Operation oder manuell aktualisieren AWS CLI. Im folgenden Beispiel wird gezeigt, wie Sie Spaltenstatistiken mithilfe von AWS CLI aktualisieren.

```
aws glue update-column-statistics-for-table --cli-input-json:

{
    "CatalogId": "111122223333",
    "DatabaseName": "database_name",
    "TableName": "table_name",
    "ColumnStatisticsList": [
        {
            "ColumnName": "col1",
            "ColumnType": "Boolean",
            "AnalyzedTime": "1970-01-01T00:00:00",
            "StatisticsData": {
                "Type": "BOOLEAN",
                "BooleanColumnStatisticsData": {
                    "NumberOfTrues": 5,
                    "NumberOfFalses": 5,
                    "NumberOfNulls": 0
                }
            }
        }
    ]
}
```

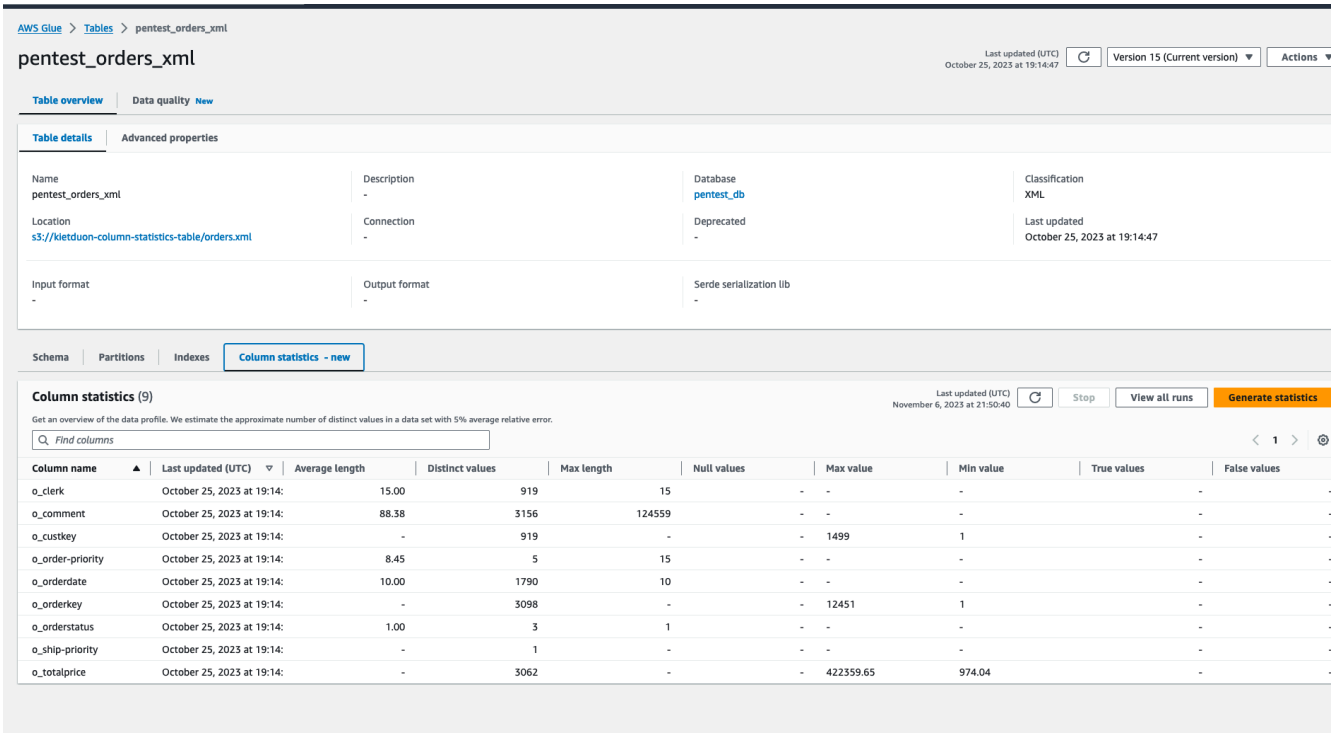
Anzeigen von Spaltenstatistiken

Nach erfolgreicher Generierung der Statistiken speichert Data Catalog diese Informationen für die kostenbasierten Optimierer in Amazon Athena und Amazon Redshift, um bei der Ausführung von Abfragen optimale Entscheidungen treffen zu können. Die Statistiken variieren je nach Spaltentyp.

AWS Management Console

Anzeigen von Spaltenstatistiken für eine Tabelle

- Nach dem Ausführen der Spaltenstatistik-Aufgabe werden in der Registerkarte Spaltenstatistiken auf der Seite Tabellendetails die Statistiken für die Tabelle angezeigt.



Die folgenden Statistiken sind verfügbar:

- Spaltenname: Der Spaltenname, der zur Generierung der Statistiken verwendet wurde.
- Letzte Aktualisierung: Das Datum und die Uhrzeit der Erstellung.
- Durchschnittliche Länge: durchschnittliche Länge der Werte in der Spalte.
- Eindeutige Werte: Gesamtzahl der unterschiedlichen Werte in der Spalte. Wir schätzen die Anzahl unterschiedlicher Werte in einer Spalte mit einer relativen Fehlerquote von 5 %.
- Höchstwert: der höchste Wert in der Spalte.
- Mindestwert: der kleinste Wert in der Spalte.
- Maximale Länge: die Länge des höchsten Werts in der Spalte.
- Null-Werte: die Anzahl der Null-Werte in der Spalte.
- „Wahr“-Werte: die Anzahl der „Wahr“-Werte in der Spalte.
- „Falsch“-Werte: die Anzahl der „Falsch“-Werte in der Spalte.

- numFiles: Die Gesamtzahl der Dateien in der Tabelle. Dieser Wert ist auf der Registerkarte Erweiterte Eigenschaften verfügbar.

AWS CLI

Im folgenden Beispiel wird gezeigt, wie Sie Spaltenstatistiken mithilfe von AWS CLI abrufen.

```
aws glue get-column-statistics-for-table \
    --database-name database_name \
    --table-name table_name \
    --column-names <column_name>
```

Sie können die Spaltenstatistiken auch mithilfe der GetColumnStatisticsForTableAPI-Operation anzeigen.

Anzeigen von Spaltenstatistik-Aufgabenläufen

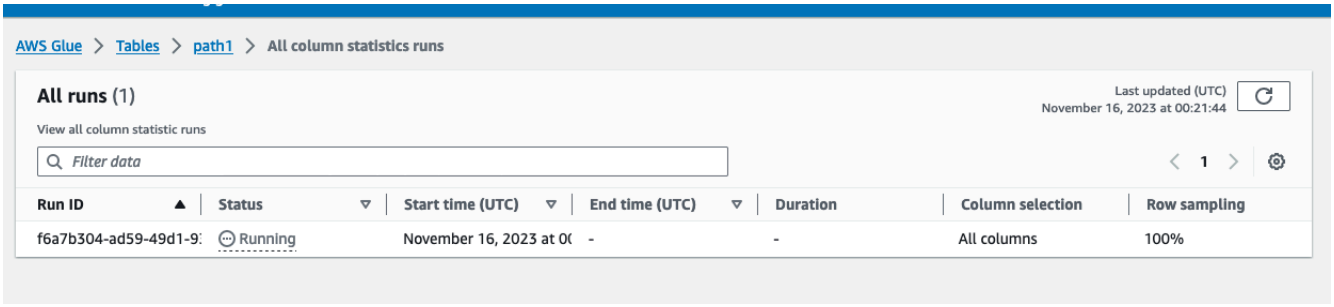
Nachdem Sie eine Aufgabe mit Spaltenstatistiken ausgeführt haben, können Sie die Details der Aufgabenausführung für eine Tabelle mithilfe der AWS Glue Konsole AWS CLI oder mithilfe von GetColumnStatisticsTaskRunsOperation untersuchen.

Console

Anzeigen von Details zur Ausführung einer Spaltenstatistik-Aufgabe

1. Wählen Sie auf der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie eine Tabelle mit Spaltenstatistiken aus.
3. Wählen Sie auf der Seite Tabellendetails die Option Spaltenstatistiken aus.
4. Wählen Sie Ausführungen anzeigen aus.

   Sie können Informationen zu allen Ausführungen in Verbindung mit der angegebenen Tabelle anzeigen.



AWS CLI

Ersetzen Sie Werte für `DatabaseName` und `TableName` im folgenden Beispiel durch den tatsächlichen Datenbank- und Tabellennamen.

```
aws glue get-column-statistics-task-runs --input-cli-json file://input.json
{
    "DatabaseName": "database_name",
    "TableName": "table_name"
}
```

Beenden des Spaltenstatistik-Aufgabenlaufs

Sie können die Ausführung einer Spaltenstatistikaufgabe für eine Tabelle mithilfe der AWS Glue Konsole AWS CLI oder mithilfe von StopColumnStatisticsTaskRunOperation beenden.

Console

Einen Spaltenstatistik-Aufgabenlauf beenden Sie wie folgt:

1. Wählen Sie in der AWS Glue Konsole unter Datenkatalog die Option Tabellen aus.
2. Wählen Sie die Tabelle mit der Spalte „Spaltenstatistik-Aufgabenlauf wird ausgeführt“ aus.
3. Wählen Sie auf der Seite Tabellendetails die Option Spaltenstatistiken aus.
4. Wählen Sie Beenden aus.

   Wenn Sie die Aufgabe beenden, bevor die Ausführung abgeschlossen ist, werden keine Spaltenstatistiken für die Tabelle generiert.

AWS CLI

Ersetzen Sie Werte für `DatabaseName` und `TableName` im folgenden Beispiel durch den tatsächlichen Datenbank- und Tabellennamen.

```
aws glue stop-column-statistics-task-run --input-cli-json file://input.json
{
    "DatabaseName": "database_name",
    "TableName": "table_name"
}
```

Löschen von Spaltenstatistiken

Sie können Spaltenstatistiken mithilfe der DeleteColumnStatisticsForTableAPI-Operation oder löschen AWS CLI. Das folgende Beispiel zeigt, wie Spaltenstatistiken mit AWS Command Line Interface (AWS CLI) gelöscht werden.

```
aws glue delete-column-statistics-for-table \
    --database-name 'database_name' \
    --table-name 'table_name' \
    --column-name 'column_name'
```

Überlegungen und Einschränkungen

Die folgenden Überlegungen und Einschränkungen gelten für die Generierung von Spaltenstatistiken.

Überlegungen

- Die Verwendung von Stichproben zur Generierung von Statistiken reduziert die Laufzeit, kann jedoch zu ungenauen Statistiken führen.
- Data Catalog speichert keine unterschiedlichen Versionen der Statistiken.
- Sie können pro Tabelle nur jeweils eine Aufgabe zur Erstellung von Statistiken gleichzeitig ausführen.
- Wenn eine Tabelle mit einem im Data Catalog registrierten AWS KMS Kundenschlüssel verschlüsselt ist, AWS Glue verwendet sie denselben Schlüssel zum Verschlüsseln von Statistiken.

Die Aufgabe für Spaltenstatistiken unterstützt das Generieren von Statistiken:

- Wenn die IAM-Rolle über vollständige Tabellenberechtigungen verfügt (IAM oder Lake Formation).
- Wenn die IAM-Rolle im Lake-Formation-Hybridzugriffsmodus über Berechtigungen für die Tabelle verfügt.

Die Aufgabe für Spaltenstatistiken unterstützt das Generieren von Statistiken in folgenden Fällen nicht:

- Tische mit zellbasierter Zutrittskontrolle von Lake Formation
- Transaktionale Datenseen — Linux-Stiftung Delta Lake, Apache Hudi
- Tabellen in Verbunddatenbanken – Hive Metastore, Amazon Redshift Datashares
- Verschachtelte Spalten, Arrays und Strukturdatentypen.
- Tabelle, die von einem anderen Konto aus mit Ihnen geteilt wurde

Verschlüsseln Ihres Data Catalog

Sie können Ihre im Ruhezustand gespeicherten Metadaten mithilfe von Verschlüsselungsschlüsseln schützen, die AWS Glue Data Catalog von AWS Key Management Service (AWS KMS) verwaltet werden. Sie können die Datenkatalogverschlüsselung für neue Datenkataloge mithilfe der Datenkatalogeinstellungen aktivieren. Sie können die Verschlüsselung für einen vorhandenen Datenkatalog nach Bedarf aktivieren oder deaktivieren. Wenn diese Option aktiviert ist, werden alle neuen Metadaten, die in den Katalog geschrieben werden, AWS Glue verschlüsselt, während vorhandene Metadaten unverschlüsselt bleiben.

Ausführliche Informationen zur Verschlüsselung Ihres Datenkatalogs finden Sie unter. Verschlüsseln Ihres Data Catalog

Sicherung Ihres Datenkatalogs mit Lake Formation

AWS Lake Formation ist ein Service, der es einfacher macht, einen sicheren Data Lake einzurichten AWS. Er bietet einen zentralen Ort, an dem Sie Ihre Data Lakes erstellen und sicher verwalten können, indem Sie detaillierte Zugriffsberechtigungen definieren. Lake Formation verwendet den Datenkatalog, um Metadaten zu Ihrem Data Lake zu speichern und abzurufen, z. B. Tabellendefinitionen, Schemainformationen und Einstellungen für die Datenzugriffskontrolle.

Sie können Ihren Amazon S3 S3-Datenspeicherort der Metadatentabelle oder Datenbank bei Lake Formation registrieren und damit Berechtigungen auf Metadatenebene für die Datenkatalogressourcen definieren. Sie können Lake Formation auch verwenden, um Speicherzugriffsberechtigungen für die zugrunde liegenden Daten zu verwalten, die in Amazon S3 im Auftrag integrierter Analyse-Engines gespeichert sind.

Weitere Informationen finden Sie unter Was ist AWS Lake Formation? .

Arbeiten mit AWS Glue Data Catalog Ansichten in AWS Glue

Sie können Ansichten in den AWS Glue Data Catalog, allgemein als AWS Glue Data Catalog Ansichten bekannt, erstellen und verwalten. Diese Ansichten sind nützlich, da sie mehrere SQL-Abfrage-Engines unterstützen, sodass Sie über verschiedene AWS Dienste hinweg auf dieselbe Ansicht zugreifen können Amazon Athena, z. B. Amazon Redshift, und AWS Glue. Sie können Ansichten verwenden, die auf Apache Iceberg, Apache Hudi und Delta Lake basieren.

Indem Sie eine Ansicht im Datenkatalog erstellen, können Sie mithilfe von Ressourcenzuweisungen und tagbasierten Zugriffskontrollen Zugriff AWS Lake Formation darauf gewähren. Mit dieser Methode der Zugriffskontrolle müssen Sie keinen zusätzlichen Zugriff auf die Tabellen konfigurieren, auf die bei der Erstellung der Ansicht verwiesen wird. Diese Methode zur Erteilung von Berechtigungen wird Definer-Semantik genannt, und diese Ansichten werden Defineransichten genannt. Weitere Informationen zur Zugriffskontrolle finden Sie unter Erteilen und Widerrufen von Berechtigungen für Datenkatalogressourcen im Entwicklerhandbuch. AWS Lake Formation AWS Lake Formation

Datenkatalogansichten sind für die folgenden Anwendungsfälle nützlich:

- Granulare Zugriffskontrolle — Sie können eine Ansicht erstellen, die den Datenzugriff auf der Grundlage der vom Benutzer benötigten Berechtigungen einschränkt. Sie können beispielsweise Ansichten im Datenkatalog verwenden, um zu verhindern, dass Mitarbeiter, die nicht in der Personalabteilung arbeiten, personenbezogene Daten (PII) sehen.
- Vollständige Definition der Ansicht — Indem Sie Filter auf Ihre Ansicht im Datenkatalog anwenden, stellen Sie sicher, dass die in der Ansicht verfügbaren Datensätze immer vollständig sind.
- Verbesserte Sicherheit — Die zur Erstellung der Ansicht verwendete Abfragedefinition muss vollständig sein, sodass Datenkatalogansichten weniger anfällig für SQL-Befehle von böswilligen Akteuren sind.
- Einfache gemeinsame Nutzung von Daten — Nutzen Sie die Möglichkeit, Daten mit anderen AWS Konten auszutauschen, ohne Daten verschieben zu müssen. AWS Lake Formation

Erstellen einer Data-Catalog-Ansicht

Sie können Datenkatalogansichten mit den AWS CLI und AWS Glue ETL-Skripts mit Spark SQL erstellen. Die Syntax für die Erstellung einer Datenkatalogsicht beinhaltet die Angabe des Ansichtstyps als `MULTI DIALECT` und des `SECURITY` Prädikats als`DEFINER`, wodurch eine definierende Ansicht angegeben wird.

Beispiel für eine SQL-Anweisung zum Erstellen einer Datenkatalogsicht:

```
CREATE PROTECTED MULTI DIALECT VIEW database_name.catalog_view SECURITY DEFINER
AS SELECT order_date, sum(totalprice) AS price
FROM source_table
GROUP BY order_date;
```

Nachdem Sie eine Datenkatalogsicht erstellt haben, können Sie eine IAM-Rolle mit der AWS Lake Formation `SELECT`-Berechtigung für die Ansicht verwenden, um sie von Diensten wie Amazon Athena Amazon Redshift, oder AWS Glue ETL-Jobs abzufragen. Sie müssen keinen Zugriff auf die zugrunde liegenden Tabellen gewähren, auf die in der Ansicht verwiesen wird.

Weitere Informationen zum Erstellen und Konfigurieren von Datenkatalogsichten finden Sie unter Erstellen von AWS Glue Datenkatalogsichten im AWS Lake Formation Entwicklerhandbuch.

Unterstützte View-Operationen

Die folgenden Befehlsfragmente zeigen Ihnen verschiedene Möglichkeiten, mit Datenkatalogsichten zu arbeiten:

ANSICHT ERSTELLEN

Erstellt eine Datenkatalogansicht. Im Folgenden finden Sie ein Beispiel, das das Erstellen einer Ansicht aus einer vorhandenen Tabelle zeigt:

```
CREATE PROTECTED MULTI DIALECT VIEW catalog_view
SECURITY DEFINER AS SELECT * FROM my_catalog.my_database.source_table
```

ANSICHT ÄNDERN

Verfügbare Syntax:

```
ALTER VIEW view_name [FORCE] ADD DIALECT AS query
ALTER VIEW view_name [FORCE] UPDATE DIALECT AS query
```

```
ALTER VIEW view_name DROP DIALECT
```

Sie können die `FORCE ADD DIALECT` Option verwenden, um die Aktualisierung des Schemas und der Unterobjekte gemäß dem neuen Engine-Dialekt zu erzwingen. Beachten Sie, dass dies zu Abfragefehlern führen kann, wenn Sie nicht auch andere Engine-Dialekte aktualisieren. `FORCE` Im Folgenden wird ein Beispiel gezeigt:

```
ALTER VIEW catalog_view FORCE ADD DIALECTAS
SELECT order_date, sum(totalprice) AS priceFROM source_tableGROUP BY orderdate;
```

Im Folgenden wird gezeigt, wie eine Ansicht geändert wird, um den Dialekt zu aktualisieren:

```
ALTER VIEW catalog_view UPDATE DIALECT AS
SELECT count(*) FROM my_catalog.my_database.source_table;
```

ANSICHT BESCHREIBEN

Verfügbare Syntax für die Beschreibung einer Ansicht:

`SHOW COLUMNS {FROM|IN} view_name [{FROM|IN} database_name]`— Wenn der Benutzer über die erforderlichen AWS Lake Formation Rechte AWS Glue und Rechte zur Beschreibung der Ansicht verfügt, kann er die Spalten auflisten. Im Folgenden finden Sie einige Beispielbefehle zum Anzeigen von Spalten:

```
SHOW COLUMNS FROM my_database.source_table;
SHOW COLUMNS IN my_database.source_table;
```

`DESCRIBE view_name`— Wenn der Benutzer über die erforderlichen AWS Lake Formation Rechte AWS Glue und Berechtigungen zur Beschreibung der Ansicht verfügt, kann er die Spalten in der Ansicht zusammen mit ihren Metadaten auflisten.

ANSICHT LÖSCHEN

Verfügbare Syntax:

```
DROP VIEW [ IF EXISTS ] view_name
```

Das folgende Beispiel zeigt eine `DROP` Anweisung, die testet, ob eine Ansicht existiert, bevor sie gelöscht wird:

```
DROP VIEW IF EXISTS catalog_view;
```

`SHOW CREATE VIEW view_name`— Zeigt die SQL-Anweisung an, die die angegebene Ansicht erstellt. Das folgende Beispiel zeigt die Erstellung einer Datenkatalogansicht:

```
SHOW CREATE TABLE my_database.catalog_view;CREATE PROTECTED MULTI DIALECT VIEW
 my_catalog.my_database.catalog_view (
  net_profit,
  customer_id,
  item_id,
  sold_date)
TBLPROPERTIES (
  'transient_lastDdlTime' = '1736267222')
SECURITY DEFINER AS SELECT * FROM
my_database.store_sales_partitioned_lf WHERE customer_id IN (SELECT customer_id from
 source_table limit 10)
```

ANSICHTEN ANZEIGEN

Listet alle Ansichten im Katalog auf, z. B. reguläre Ansichten, Ansichten mit mehreren Dialekten (MDV) und MDV ohne Spark-Dialekt. Die verfügbare Syntax lautet wie folgt:

```
SHOW VIEWS [{ FROM | IN } database_name] [LIKE regex_pattern]:
```

Im Folgenden wird ein Beispielbefehl zum Anzeigen von Ansichten gezeigt:

```
SHOW VIEWS IN marketing_analytics LIKE 'catalog_view*';
```

Weitere Informationen zum Erstellen und Konfigurieren von Datenkatalogansichten finden Sie unter Erstellen von AWS Glue Datenkatalogsichten im AWS Lake Formation Entwicklerhandbuch.

Abfrage einer Data-Catalog-Ansicht

Nachdem Sie eine Datenkatalogansicht erstellt haben, können Sie die Ansicht abfragen. Die in Ihren AWS Glue Jobs konfigurierte IAM-Rolle muss über die Lake Formation SELECT-Berechtigung in der Datenkatalogansicht verfügen. Sie müssen keinen Zugriff auf die zugrunde liegenden Tabellen gewähren, auf die in der Ansicht verwiesen wird.

Sobald Sie alles eingerichtet haben, können Sie Ihre Ansicht abfragen. Sie können beispielsweise die folgende Abfrage ausführen, um auf eine Ansicht zuzugreifen.

```
SELECT * from my_database.catalog_view LIMIT 10;
```

Einschränkungen

Beachten Sie bei der Verwendung von Datenkatalogansichten die folgenden Einschränkungen.

- Sie können Datenkatalogsichten nur mit AWS Glue Version 5.0 und höher erstellen.
- Der Data Catalog View Definer muss `SELECT` Zugriff auf die zugrunde liegenden Basistabellen haben, auf die über die Ansicht zugegriffen wird. Das Erstellen der Datenkatalogansicht schlägt fehl, wenn einer bestimmten Basistabelle Lake Formation-Filter zugewiesen wurden, die der Definiererrolle zugewiesen wurden.
- Basistabellen dürfen nicht über die `IAMAllowedPrincipals` Data Lake-Berechtigung verfügen. AWS Lake Formation Falls vorhanden, tritt der Fehler Multi Dialect views may only reference tables without IAMAllowed Principals permissions auf.
- Der Amazon S3-Standort der Tabelle muss als AWS Lake Formation Data Lake-Standort registriert sein. Wenn die Tabelle nicht registriert ist, `Multi Dialect views may only reference AWS Lake Formation managed tables` tritt der Fehler auf. Informationen zur Registrierung von Amazon Amazon S3 S3-Standorten finden Sie unter Registrierung eines Amazon S3 S3-Standorts im AWS Lake Formation Entwicklerhandbuch. AWS Lake Formation
- Sie können nur `PROTECTED` Datenkatalogansichten erstellen. `UNPROTECTED`Ansichten werden nicht unterstützt.
- Sie können in einer Datenkatalog-Ansichtsdefinition nicht auf Tabellen in einem anderen AWS Konto verweisen. Sie können auch nicht auf eine Tabelle in demselben Konto verweisen, das sich in einer separaten Region befindet.
- Um Daten konto- oder regionsübergreifend gemeinsam zu nutzen, muss die gesamte Ansicht mithilfe von AWS Lake Formation Ressourcenlinks konto- und regionsübergreifend gemeinsam genutzt werden.
- Benutzerdefinierte Funktionen (UDFs) werden nicht unterstützt.
- Sie können in Datenkatalogansichten nicht auf andere Ansichten verweisen.

Zugriff auf den Datenkatalog

Sie können den AWS Glue Data Catalog (Datenkatalog) verwenden, um Ihre Daten zu entdecken und zu verstehen. Der Datenkatalog bietet eine konsistente Methode zur Verwaltung von

Schemadefinitionen, Datentypen, Speicherorten und anderen Metadaten. Sie können mit den folgenden Methoden auf den Datenkatalog zugreifen:

- AWS Glue Konsole — Sie können über die AWS Glue Konsole, eine webbasierte Benutzeroberfläche, auf den Datenkatalog zugreifen und ihn verwalten. Die Konsole ermöglicht Ihnen das Durchsuchen und Suchen nach Datenbanken, Tabellen und den zugehörigen Metadaten sowie das Erstellen, Aktualisieren und Löschen von Metadatendefinitionen.
- AWS-Glue-Crawler — Crawler sind Programme, die Ihre Datenquellen automatisch scannen und den Datenkatalog mit Metadaten füllen. Sie können Crawler erstellen und ausführen, um Daten aus verschiedenen Quellen wie Amazon S3, Amazon RDS, Amazon DynamoDB und JDBC-kompatiblen relationalen Datenbanken wie MySQL und PostgreSQL sowie aus mehreren Nichtquellen wie Snowflake und Google zu entdecken und zu katalogisieren. Amazon CloudWatchAWS BigQuery
- AWS Glue APIs — Sie können programmgesteuert auf den Datenkatalog zugreifen, indem Sie AWS Glue APIs Diese APIs ermöglichen Ihnen die programmgesteuerte Interaktion mit dem Datenkatalog und ermöglichen so die Automatisierung und Integration mit anderen Anwendungen und Diensten.
- AWS Command Line Interface (AWS CLI) — Sie können den verwenden AWS CLI , um über die Befehlszeile auf den Datenkatalog zuzugreifen und ihn zu verwalten. Die CLI bietet Befehle zum Erstellen, Aktualisieren und Löschen von Metadatendefinitionen sowie zum Abfragen und Abrufen von Metadateninformationen.
- Integration mit anderen AWS Diensten — Der Datenkatalog lässt sich in verschiedene andere AWS Dienste integrieren, sodass Sie auf die im Katalog gespeicherten Metadaten zugreifen und diese nutzen können. Sie können Amazon Athena beispielsweise verwenden, um Datenquellen mithilfe der Metadaten im Datenkatalog abzufragen und den Datenzugriff und die Datenverwaltung für die Datenkatalogressourcen AWS Lake Formation zu verwalten.

Themen

- Mit dem AWS Glue Iceberg-REST-Endpunkt eine Verbindung zum Datenkatalog herstellen
- Verbindung zum Datenkatalog mithilfe des AWS Glue Iceberg-REST-Erweiterungsendpunkts herstellen
- AWS Glue REST APIs für Apache Iceberg-Spezifikationen
- Von einer eigenständigen Spark-Anwendung aus eine Verbindung zum Datenkatalog herstellen
- Datenzuordnung zwischen Amazon Redshift und Apache Iceberg

- Überlegungen und Einschränkungen bei der Verwendung von AWS Glue Iceberg REST Catalog APIs

Mit dem AWS Glue Iceberg-REST-Endpunkt eine Verbindung zum Datenkatalog herstellen

AWS Glue Der Iceberg-REST-Endpunkt unterstützt API-Operationen, die in der Apache Iceberg REST-Spezifikation spezifiziert sind. Mithilfe eines Iceberg-REST-Clients können Sie Ihre Anwendung, die auf einer Analytics-Engine ausgeführt wird, mit dem im Datenkatalog gehosteten REST-Katalog verbinden.

Der Endpunkt unterstützt beide Apache Iceberg-Tabellenspezifikationen — v1 und v2, wobei standardmäßig v2 verwendet wird. Wenn Sie die Iceberg-Tabellenspezifikation v1 verwenden, müssen Sie v1 im API-Aufruf angeben. Mithilfe der API-Operation können Sie auf Iceberg-Tabellen zugreifen, die sowohl im Amazon S3 S3-Objektspeicher als auch im Amazon S3 S3-Tabellenspeicher gespeichert sind.

Konfiguration des Endpunkts

Sie können über den Service-Endpunkt auf den AWS Glue Iceberg-REST-Katalog zugreifen. Informationen zum regionsspezifischen AWS Glue Endpunkt finden Sie im Referenzhandbuch für Service-Endpunkte. Wenn Sie beispielsweise eine Verbindung mit der AWS Glue Region us-east-1 herstellen, müssen Sie die Eigenschaft Endpunkt-URI wie folgt konfigurieren:

```
Endpoint : https://glue.us-east-1.amazonaws.com/iceberg
```

Zusätzliche Konfigurationseigenschaften — Wenn Sie den Iceberg-Client verwenden, um eine Analyse-Engine wie Spark mit dem Service-Endpunkt zu verbinden, müssen Sie die folgenden Eigenschaften für die Anwendungskonfiguration angeben:

```
catalog_name = "mydatacatalog"
aws_account_id = "123456789012"
aws_region = "us-east-1"
spark = SparkSession.builder \
    ... \
    .config("spark.sql.defaultCatalog", catalog_name) \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog") \
```

```
    .config(f"spark.sql.catalog.{catalog_name}.type", "rest") \
    .config(f"spark.sql.catalog.{catalog_name}.uri", "https://glue.
{aws_region}.amazonaws.com/iceberg") \
    .config(f"spark.sql.catalog.{catalog_name}.warehouse", "{aws_account_id}") \
    .config(f"spark.sql.catalog.{catalog_name}.rest.sigv4-enabled", "true") \
    .config(f"spark.sql.catalog.{catalog_name}.rest.signing-name", "glue") \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .getOrCreate()
```

AWS Glue Der Iceberg-Endpunkt `https://glue.us-east-1.amazonaws.com/iceberg` unterstützt den folgenden Iceberg-REST: APIs

- GetConfig
- ListNamespaces
- CreateNamespace
- LoadNamespaceMetadata
- UpdateNamespaceProperties
- DeleteNamespace
- ListTables
- CreateTable
- LoadTable
- TableExists
- UpdateTable
- DeleteTable

Parameter für Präfix und Katalogpfad

Der Iceberg-REST-Katalog APIs enthält in seiner Anfrage ein Präfix in freier Form. URLs Beispielsweise verwendet der `ListNamespaces` API-Aufruf das `GET/v1/{prefix}/namespaces` URL-Format. AWS Glue Das Präfix folgt immer der `/catalogs/{catalog}` Struktur, um sicherzustellen, dass der REST-Pfad der Hierarchie mit AWS Glue mehreren Katalogen entspricht. Der `{catalog}` Pfadparameter kann auf der Grundlage der folgenden Regeln abgeleitet werden:

Zugriffsmuster	Kleberkatalog ID Style	Präfix-Stil	Beispiel für eine Standard-Katalog-ID	Beispiel für eine REST-Route
Greifen Sie im Girokonto auf den Standardk atalog zu	nicht erforderlich	:	n.v.	GET /v1/catal ogs/:/nam espaces
Greifen Sie in einem bestimmte n Konto auf den Standardkatalog zu	accountID	accountID	111122223333	GET /v1/catal ogs/11112 2223333/n amespaces
Greifen Sie im Girokonto auf einen verschach telten Katalog zu	katalog1/ katalog2	katalog1/ katalog2	rms-Katal og1:db1	GET /v1/catal ogs/rmsca talog1:db1/ namespaces
Greifen Sie in einem bestimmte n Konto auf einen verschach telten Katalog zu	Konto-ID: Katalog1/ Katalog2	Konto-ID: Katalog1/ Katalog2	123456789012/ rms-Katalog1: db1	GET /v1/catal ogs/12345 6789012:r mscatalog1:db1/ namespaces

Diese Zuordnung von Katalog-ID zu Präfix ist nur erforderlich, wenn Sie den REST direkt aufrufen. APIs Wenn Sie mit dem AWS Glue Iceberg-REST-Katalog APIs über eine Engine arbeiten, müssen Sie die AWS Glue Katalog-ID im `warehouse` Parameter für Ihre Iceberg-REST-Katalog-API-Einstellung oder im `glue.id` Parameter für Ihre AWS Glue Erweiterungs-API-Einstellung angeben. Wie Sie ihn beispielsweise mit EMR Spark verwenden können, erfahren Sie unter Verwenden eines Eisberg-Clusters mit Spark.

Namespace-Pfadparameter

Namespaces im APIs Iceberg-REST-Katalogpfad können mehrere Ebenen haben. Unterstützt jedoch AWS Glue nur Namespaces mit einer Ebene. Um auf einen Namespace in einer

mehrstufigen Kataloghierarchie zuzugreifen, können Sie eine Verbindung zu einem mehrstufigen Katalog über dem Namespace herstellen, um auf den Namespace zu verweisen. Auf diese Weise kann jede Abfrageengine, die die dreiteilige Notation von unterstützt, im Vergleich `catalog.namespace.table` zur Verwendung des mehrstufigen AWS Glue Namespaces ohne Kompatibilitätsprobleme auf Objekte in der mehrstufigen Kataloghierarchie zugreifen.

Verbindung zum Datenkatalog mithilfe des AWS Glue Iceberg-REST-Erweiterungsendpunkts herstellen

AWS Glue Der Iceberg-REST-Erweiterungsendpunkt bietet zusätzliche Funktionen APIs, die nicht in der Apache Iceberg-REST-Spezifikation enthalten sind, und bietet serverseitige Scanplanungsfunktionen. Diese zusätzlichen APIs werden verwendet, wenn Sie auf Tabellen zugreifen, die im verwalteten Amazon Redshift Redshift-Speicher gespeichert sind. Auf den Endpunkt kann von einer Anwendung aus zugegriffen werden, die Apache Iceberg-Erweiterungen AWS Glue Data Catalog verwendet.

Endpunktkonfiguration — Über den Service-Endpunkt kann auf einen Katalog mit Tabellen im von Redshift verwalteten Speicher zugegriffen werden. Informationen zum regionsspezifischen AWS Glue Endpunkt finden Sie im Referenzhandbuch für Service-Endgeräte. Wenn Sie beispielsweise eine Verbindung mit der AWS Glue Region us-east-1 herstellen, müssen Sie die Eigenschaft Endpunkt-URI wie folgt konfigurieren:

```
Endpoint : https://glue.us-east-1.amazonaws.com/extensions
```

```
catalog_name = "myredshiftcatalog"
aws_account_id = "123456789012"
aws_region = "us-east-1"
spark = SparkSession.builder \
    .config("spark.sql.defaultCatalog", catalog_name) \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.type", "glue") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.id",
 "{123456789012}:redshiftnamespacecatalog/redshiftdb") \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .getOrCreate()
```

AWS Glue REST APIs für Apache Iceberg-Spezifikationen

Dieser Abschnitt enthält Spezifikationen zum AWS Glue Iceberg-REST-Katalog und zur AWS Glue REST-Erweiterung APIs sowie Überlegungen zu deren Verwendung. APIs

API-Anfragen an die AWS Glue Data Catalog Endgeräte werden mit AWS Signature Version 4 (Sigv4) authentifiziert. Weitere Informationen zu AWS Sigv4 finden Sie im Abschnitt Signature Version 4 für API-Anfragen. AWS

Beim Zugriff auf den AWS Glue Dienstendpunkt und die AWS Glue Metadaten nimmt die Anwendung eine IAM-Rolle ein, für die eine IAM-Aktion erforderlich ist`glue:getCatalog`.

Der Zugriff auf den Datenkatalog und seine Objekte kann mithilfe von IAM-, Lake Formation- oder Lake Formation Formation-Hybridmodus-Berechtigungen verwaltet werden.

Verbundkataloge im Datenkatalog haben in Lake Formation registrierte Datenspeicherorte. Lake Formation arbeitet mit dem Datenkatalog zusammen, um Datenbankberechtigungen zur Verwaltung des Benutzerzugriffs auf Datenkatalogobjekte bereitzustellen.

Sie können IAM- oder Lake Formation Formation-Hybridmodus-Berechtigungen verwenden AWS Lake Formation, um den Zugriff auf den Standard-Datenkatalog und seine Objekte zu verwalten.

Um Daten in verwalteten Objekten von Lake Formation zu erstellen, einzufügen oder zu löschen, müssen Sie spezifische Berechtigungen für den IAM-Benutzer oder die IAM-Rolle einrichten.

- CREATE_CATALOG — Erforderlich, um Kataloge zu erstellen
- CREATE_DATABASE — Erforderlich, um Datenbanken zu erstellen
- CREATE_TABLE — Erforderlich, um Tabellen zu erstellen
- DELETE — Erforderlich, um Daten aus einer Tabelle zu löschen
- DESCRIBE — Erforderlich, um Metadaten zu lesen
- DROP — Erforderlich, um eine Tabelle oder Datenbank zu löschen/zu löschen
- INSERT — Wird benötigt, wenn der Principal Daten in eine Tabelle einfügen muss
- SELECT — Wird benötigt, wenn der Principal Daten aus einer Tabelle auswählen muss

Weitere Informationen finden Sie in der Berechtigungsreferenz für Lake Formation im AWS Lake Formation Entwicklerhandbuch.

GetConfig

Allgemeine Informationen

Name der Operation	GetConfig
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET /iceberg/v1/config
IAM-Aktion	kleben: GetCatalog
Genehmigungen für Lake Formation	Nicht zutreffend
CloudTrail event	Klebstoff: GetCatalog
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L67

Überlegungen und Einschränkungen

- Der `warehouse` Abfrageparameter muss auf die AWS Glue Katalog-ID festgelegt werden. Wenn nicht festgelegt, wird der Stammkatalog im aktuellen Konto verwendet, um die Antwort zurückzugeben. Weitere Informationen finden Sie unter the section called “Parameter für Präfix und Katalogpfad”.

GetCatalog

Allgemeine Informationen

Name der Operation	GetCatalog
Typ	AWS Glue Erweiterungs-API
REST-Pfad	GET/extensions/v1/catalogs/{Katalog}
IAM-Aktion	Klebstoff: GetCatalog

Genehmigungen für Lake Formation	DESCRIBE
CloudTrail event	Klebstoff: GetCatalog
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L40

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.

ListNamespaces

Allgemeine Informationen

Name der Operation	ListNamespaces
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET/iceberg/v1/catalogs/{catalog}/namespaces
IAM-Aktion	kleben: GetDatabase
Genehmigungen für Lake Formation	ALLES, BESCHREIBE, WÄHLE
CloudTrail event	kleben: GetDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L205

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.

- Es werden nur Namespaces der nächsten Ebene angezeigt. Um Namespaces auf tieferen Ebenen aufzulisten, geben Sie die ID des verschachtelten Katalogs im Katalogpfadparameter an.

CreateNamespace

Allgemeine Informationen

Name der Operation	CreateNamespace
Typ	Iceberg REST-Katalog-API
REST-Pfad	POST/iceberg/v1/catalogs/{catalog}/namespaces
IAM-Aktion	kleben: CreateDatabase
Genehmigungen für Lake Formation	ALLES, BESCHREIBE, WÄHLE
CloudTrail event	kleben: CreateDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L256

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Es kann nur ein Namespace mit einer Ebene erstellt werden. Um einen Namespace mit mehreren Ebenen zu erstellen, müssen Sie jede Ebene iterativ erstellen und mithilfe des Katalogpfadparameters eine Verbindung zu der Ebene herstellen.

StartCreateNamespaceTransaction

Allgemeine Informationen

Name der Operation	StartCreateNamespaceTransaction
Typ	AWS Glue Erweiterungs-API

REST-Pfad	POST/extensions/v1/catalogs/{catalog}/namespaces
IAM-Aktion	kleben: CreateDatabase
Genehmigungen für Lake Formation	ALLES, BESCHREIBE, WÄHLE
CloudTrail event	kleben: CreateDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L256

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können nur einen Namespace mit einer Ebene erstellen. Um Namespaces mit mehreren Ebenen zu erstellen, müssen Sie jede Ebene iterativ erstellen und mithilfe des Katalogpfadparameters eine Verbindung zu der Ebene herstellen.
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie für die Nachverfolgung mithilfe des API-Aufrufs verwenden können. `CheckTransactionStatus`
- Sie können diese API nur aufrufen, wenn der `GetCatalog` API-Aufruf den Parameter `use-extensions=true` in der Antwort enthält.

LoadNamespaceMetadata

Allgemeine Informationen

Name des Vorgangs	LoadNamespaceMetadata
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET/iceberg/v1/catalogs/{catalog}/namespaces/{ns}
IAM-Aktion	kleben: GetDatabase
Genehmigungen für Lake Formation	ALLES, BESCHREIBE, WÄHLE

CloudTrail event	kleben: GetDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L302

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Pfadparameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

UpdateNamespaceProperties

Allgemeine Informationen

Name des Vorgangs	UpdateNamespaceProperties
Typ	Iceberg REST-Katalog-API
REST-Pfad	POST /iceberg/v1/catalogs/{catalog}/namespaces/{ns}/properties
IAM-Aktion	kleben: UpdateDatabase
Genehmigungen für Lake Formation	RUF AN, SPÄTER
CloudTrail event	kleben: UpdateDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L400

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Pfadparameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

DeleteNamespace

Allgemeine Informationen

Name der Operation	DeleteNamespace
Typ	Iceberg REST-Katalog-API
REST-Pfad	DELETE/iceberg/v1/catalogs/{catalog}/namespces/{ns}
IAM-Aktion	kleben: DeleteDatabase
Genehmigungen für Lake Formation	ALLES, FALLENLASSEN
CloudTrail event	kleben: DeleteDatabase
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L365

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Wenn sich Objekte in der Datenbank befinden, schlägt der Vorgang fehl.
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie mithilfe des `CheckTransactionStatus` API-Aufrufs für die Nachverfolgung verwenden können.
- Die API kann nur verwendet werden, wenn der `GetCatalog` API-Aufruf als Antwort darauf hinweist`use-extensions=true`.

StartDeleteNamespaceTransaction

Allgemeine Informationen

Name der Operation	StartDeleteNamespaceTransaction

Typ	AWS Glue Erweiterungs-API
REST-Pfad	DELETE /extensions/v1/catalogs/{catalog}/namespces/{ns}
IAM-Aktion	kleben: DeleteDatabase
Genehmigungen für Lake Formation	ALLES, FALLENLASSEN
CloudTrail event	kleben: DeleteDatabase
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L85

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Wenn sich Objekte in der Datenbank befinden, schlägt der Vorgang fehl.
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie mithilfe des `CheckTransactionStatus` API-Aufrufs für die Nachverfolgung verwenden können.
- Die API kann nur verwendet werden, wenn der `GetCatalog` API-Aufruf als Antwort darauf hinweist`use-extensions=true`.

ListTables

Allgemeine Informationen

Name der Operation	ListTables
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET /iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables
IAM-Aktion	kleben: GetTables

Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail event	kleben: GetTables
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L463

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Alle Tabellen, auch Tabellen, die nicht zu ICEBERG gehören, werden aufgelistet. Um festzustellen, ob eine Tabelle als Iceberg-Tabelle geladen werden kann oder nicht, rufen Sie operation auf. `LoadTable`

CreateTable

Allgemeine Informationen

Name der Operation	CreateTable
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET /iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables
IAM-Aktion	kleben: CreateTable
Genehmigungen für Lake Formation	ALLE, CREATE_TABLE
CloudTrail event	kleben: CreateTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L497

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- `CreateTable`mit Staging wird nicht unterstützt. Wenn der `stageCreate` Abfrageparameter angegeben ist, schlägt der Vorgang fehl. Das bedeutet, dass Operation like `CREATE TABLE AS SELECT` nicht unterstützt wird und Sie eine Kombination aus `CREATE TABLE` und `INSERT INTO` als Workaround verwenden können.
- Der `CreateTable` API-Vorgang unterstützt die Option nicht. `state-create = TRUE`

StartCreateTableTransaction

Allgemeine Informationen

Name der Operation	CreateTable
Typ	AWS Glue Erweiterungs-API
REST-Pfad	POST/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables
IAM-Aktion	kleben: CreateTable
Genehmigungen für Lake Formation	ALLE, CREATE_TABLE
CloudTrail event	kleben: CreateTable
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L107

Überlegungen und Einschränkungen

- Der Katalogpfadparameter muss dem im the section called “Parameter für Präfix und Katalogpfad” Abschnitt beschriebenen Stil entsprechen.

- Sie können im REST-Pfadparameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- `CreateTable`mit Staging wird nicht unterstützt. Wenn der `stageCreate` Abfrageparameter angegeben ist, schlägt der Vorgang fehl. Das bedeutet, dass die Operation like nicht unterstützt `CREATE TABLE AS SELECT` wird und der Benutzer eine Kombination aus und verwenden sollte, um das Problem zu umgehen`CREATE TABLE`. `INSERT INTO`
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie für die Nachverfolgung mithilfe des API-Aufrufs verwenden können. `CheckTransactionStatus`
- Die API kann nur verwendet werden, wenn der `GetCatalog` API-Aufruf als Antwort darauf hinweist`use-extensions=true`.

LoadTable

Allgemeine Informationen

Vorgangsname	LoadTable
Typ	Iceberg REST-Katalog-API
REST-Pfad	GET/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: GE TTable
Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail Ereignis	kleben: GetTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L616

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen Namespace mit einer einzigen Ebene angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

- `CreateTable`mit Staging wird nicht unterstützt. Wenn der `stageCreate` Abfrageparameter angegeben ist, schlägt der Vorgang fehl. Das bedeutet, dass die Operation like nicht unterstützt `CREATE TABLE AS SELECT` wird und der Benutzer eine Kombination aus und verwenden sollte, um das Problem zu umgehen`CREATE TABLE`. `INSERT INTO`
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie für die Nachverfolgung mithilfe des API-Aufrufs verwenden können. `CheckTransactionStatus`
- Die API kann nur verwendet werden, wenn der `GetCatalog` API-Aufruf als Antwort darauf hinweist`use-extensions=true`.

ExtendedLoadTable

Allgemeine Informationen

Vorgangsname	LoadTable
Typ	AWS Glue Erweiterungs-API
REST-Pfad	GET /extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: GetTable
Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail Ereignis	kleben: GetTable
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L134

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Für den Abfrageparameter Snapshots wird nur der `all` Modus unterstützt.

- Im Vergleich zur `LoadTable` API unterscheidet sich die `ExtendedLoadTable` API in folgenden Punkten:
  - Erzwingt nicht unbedingt, dass alle Felder verfügbar sind.
  - stellt die folgenden zusätzlichen Parameter im Konfigurationsfeld der Antwort bereit:

    Zusätzliche Parameter

Konfigurationsschlüssel	Beschreibung
aws. server-side-capabilities.scan-Planung	Gibt an, ob die Tabelle mit der Tabelle und gescannt werden kann. PreparePlan PlanTable APIs
war. server-side-capabilities.data-commit	Gibt an, ob die Tabelle mithilfe der Transaktion festgeschrieben werden kann. StartUpdateTable
aws.glue.staging.location	Wird für die serverseitige Scanplanung oder Datenübergabe verwendet. Dabei handelt es sich um einen vom Service verwalteten Staging-Speicherort, der für das Modul zum Schreiben temporärer Datendateien verwendet werden kann
aws.glue.staging. access-key-id	Wird für die serverseitige Scanplanung oder Datenübergabe verwendet und ist ein Teil der temporären AWS Anmeldeinformationen für den Zugriff auf den vom Service verwalteten Staging-Speicherort
aws.glue.staging. secret-access-key	Wird für die serverseitige Scanplanung oder Datenübergabe verwendet und ist ein Teil der temporären AWS Anmeldeinformationen für den Zugriff auf den vom Service verwalteten Staging-Speicherort.
aws.glue.staging.session-token	Wird für die serverseitige Scanplanung oder Datenübergabe verwendet und ist ein Teil der temporären Anmeldeinformationen für den Zugriff auf den vom Service verwalteten Staging-Speicherort AWS .

Konfigurationsschlüssel	Beschreibung
aws.glue.staging.expiration-ms	Wird für die serverseitige Scanplanung oder die Datenüber gabe verwendet, Ablaufzeit der Anmeldeinformationen für den Zugriff auf das vom Service verwaltete Staging.
aws.glue.staging. data-transfer-role-arn	Wird für die serverseitige Scanplanung oder Datenüber gabe verwendet, eine IAM-Rolle, von der angenommen werden kann, dass sie auf den vom Service verwalteten Staging-Speicherort zugreift.

PreplanTable

Allgemeine Informationen

Vorgangsname	PreplanTable
Typ	AWS Glue Erweiterungs-API
REST-Pfad	POST /extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}/preplan
IAM-Aktion	kleben: GetTable
Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail Ereignis	kleben: GetTable
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L211

Überlegungen

- Der Katalogpfadparameter sollte dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

- Der Aufrufer dieser API sollte immer anhand des Seitentokens feststellen, ob noch Ergebnisse abgerufen werden müssen. Eine Antwort mit leerem Seitenelement, aber einem Paginierungstoken ist möglich, wenn die Serverseite noch verarbeitet, aber in der angegebenen Antwortzeit kein Ergebnis erzielen kann.
- Sie können diese API nur verwenden, wenn die `ExtendedLoadTable` API-Antwort Folgendes enthält`aws.server-side-capabilities.scan-planning=true`:

PlanTable

Allgemeine Informationen

Vorgangsname	PlanTable
Typ	AWS Glue Erweiterungs-API
REST-Pfad	POST /extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}/plan
IAM-Aktion	kleben: GetTable
Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail Ereignis	kleben: GetTable
API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L243

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Der Aufrufer dieser API sollte immer anhand des Seitentokens feststellen, ob noch Ergebnisse abgerufen werden müssen. Eine Antwort mit leerem Seitenelement, aber einem Paginierungstoken ist möglich, wenn die Serverseite noch verarbeitet, aber in der angegebenen Antwortzeit kein Ergebnis erzielen kann.

- Sie können diese API nur verwenden, wenn die `ExtendedLoadTable` API-Antwort Folgendes enthält`aws.server-side-capabilities.scan-planning=true`.

TableExists

Allgemeine Informationen

Vorgangsname	TableExists
Typ	Iceberg REST-Katalog-API
REST-Pfad	HEAD/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: GetTable
Genehmigungen für Lake Formation	ANRUFEN, AUSWÄHLEN, BESCHREIBEN
CloudTrail Ereignis	kleben: GetTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L833

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

UpdateTable

Allgemeine Informationen

Vorgangsname	UpdateTable
Typ	Iceberg REST-Katalog-API

REST-Pfad	POST /iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: UpdateTable
Genehmigungen für Lake Formation	ANRUFEN, SPÄTER
CloudTrail Ereignis	kleben: UpdateTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L677

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.

StartUpdateTableTransaction

Allgemeine Informationen

Vorgangsname	StartUpdateTableTransaction
Typ	AWS Glue Erweiterungs-API
REST-Pfad	POST/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: UpdateTable
Genehmigungen für Lake Formation	RUF AN, SPÄTER
CloudTrail Ereignis	kleben: UpdateTable

API-Definition öffnen	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L154

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- Die API ist asynchron und gibt eine Transaktions-ID zurück, die Sie für die Nachverfolgung mithilfe des `CheckTransactionStatus` API-Aufrufs verwenden können.
- Ein `RenamTable` Vorgang kann auch über diese API ausgeführt werden. In diesem Fall muss der Aufrufer auch über die Glue: - CreateTable oder LakeFormation CREATE_TABLE-Berechtigung verfügen, damit die Tabelle umbenannt werden kann.
- Sie können diese API nur verwenden, wenn die `ExtendedLoadTable` API-Antwort Folgendes enthält: `aws.server-side-capabilities.scan-planning=true`

DeleteTable

Allgemeine Informationen

Vorgangsname	DeleteTable
Typ	Iceberg REST-Katalog-API
REST-Pfad	DELETE/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: DeleteTable
Genehmigungen für Lake Formation	ALLES, FALLENLASSEN
CloudTrail Ereignis	kleben: DeleteTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L793

Überlegungen

- Der Parameter für den Katalogpfad sollte dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- `DeleteTable`Der API-Betrieb unterstützt eine Bereinigungsoption. Wenn Purge auf eingestellt ist`true`, werden die Tabellendaten gelöscht, andernfalls werden die Daten nicht gelöscht. Bei Tabellen in Amazon S3 löscht der Vorgang keine Tabellendaten. Der Vorgang schlägt fehl, wenn die Tabelle in Amazon S3 gespeichert ist, und`purge = TRUE,`.

  Bei Tabellen, die im verwalteten Amazon Redshift-Speicher gespeichert sind, löscht der Vorgang Tabellendaten, ähnlich dem `DROP TABLE` Verhalten in Amazon Redshift. Der Vorgang schlägt fehl, wenn die Tabelle in Amazon Redshift und `purge = FALSE` gespeichert ist.
- `purgeRequest=true` wird nicht unterstützt.

StartDeleteTableTransaction

Allgemeine Informationen

Vorgangsname	StartDeleteTableTransaction
Typ	AWS Glue Erweiterungs-API
REST-Pfad	DELETE /extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{Tabelle}
IAM-Aktion	Klebstoff: DeleteTable
Genehmigungen für Lake Formation	ALLES, FALLENLASSEN
CloudTrail Ereignis	kleben: DeleteTable
API-Definition öffnen	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L793

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”
- Sie können im REST-Path-Parameter nur einen einstufigen Namespace angeben. Weitere Informationen finden Sie im the section called “Namespace-Pfadparameter” Abschnitt.
- `purgeRequest=false` wird nicht unterstützt.
- Die API ist asynchron und gibt eine Transaktions-ID zurück, über `CheckTransactionStatus` die nachverfolgt werden kann.

CheckTransactionStatus

Allgemeine Informationen

Vorgangsname	CheckTransactionStatus
Typ	AWS Glue Erweiterungs-API
REST-Pfad	POST/extensions/v1/transactions/status
IAM-Aktion	Dieselbe Berechtigung wie die Aktion, die die Transaktion initiiert
Genehmigungen für Lake Formation	Dieselbe Berechtigung wie die Aktion, die die Transaktion initiiert
Öffnen Sie die API-Definition	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L273

Überlegungen

- Der Katalogpfadparameter muss dem im Abschnitt beschriebenen Stil entsprechen. the section called “Parameter für Präfix und Katalogpfad”

Von einer eigenständigen Spark-Anwendung aus eine Verbindung zum Datenkatalog herstellen

Sie können von einer Standanwendung aus mithilfe eines Apache Iceberg-Connectors eine Verbindung zum Datenkatalog herstellen.

1. Erstellen Sie eine IAM-Rolle für die Spark-Anwendung.
2. Connect Sie mithilfe des AWS Glue Iceberg-Connectors eine Verbindung zum Iceberg Rest-Endpunkt her.

```
# configure your application. Refer to https://docs.aws.amazon.com/cli/latest/
userguide/cli-configure-envvars.html for best practices on configuring environment
 variables.
export AWS_ACCESS_KEY_ID=$(aws configure get appUser.aws_access_key_id)
export AWS_SECRET_ACCESS_KEY=$(aws configure get appUser.aws_secret_access_key)
export AWS_SESSION_TOKEN=$(aws configure get appUser.aws_secret_token)

export AWS_REGION=us-east-1
export REGION=us-east-1
export AWS_ACCOUNT_ID = {specify your aws account id here}

~/spark-3.5.3-bin-hadoop3/bin/spark-shell \
    --packages org.apache.iceberg:iceberg-spark-runtime-3.4_2.12:1.6.0 \
    --conf
 "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions"
 \
    --conf "spark.sql.defaultCatalog=spark_catalog" \
    --conf "spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkCatalog" \
    --conf "spark.sql.catalog.spark_catalog.type=rest" \
    --conf "spark.sql.catalog.spark_catalog.uri=https://glue.us-east-1.amazonaws.com/
iceberg" \
    --conf "spark.sql.catalog.spark_catalog.warehouse = {AWS_ACCOUNT_ID}" \
    --conf "spark.sql.catalog.spark_catalog.rest.sigv4-enabled=true" \
    --conf "spark.sql.catalog.spark_catalog.rest.signing-name=glue" \
    --conf "spark.sql.catalog.spark_catalog.rest.signing-region=us-east-1" \
    --conf "spark.sql.catalog.spark_catalog.io-
impl=org.apache.iceberg.aws.s3.S3FileIO" \
    --conf
 "spark.hadoop.fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.SimpleAWSCredentialPr
```

3. Daten im Datenkatalog abfragen.

```
spark.sql("create database myicebergdb").show()
spark.sql("""CREATE TABLE myicebergdb.mytbl (name string) USING iceberg location
 's3://bucket_name/mytbl'""")
spark.sql("insert into myicebergdb.mytbl values('demo') ").show()
```

Datenzuordnung zwischen Amazon Redshift und Apache Iceberg

Redshift und Iceberg unterstützen verschiedene Datentypen. In der folgenden Kompatibilitätsmatrix werden die Unterstützung und die Einschränkungen bei der Zuordnung von Daten zwischen diesen beiden Datensystemen beschrieben. Weitere Informationen zu den unterstützten Datentypen in den jeweiligen Datensystemen finden Sie in den Amazon Redshift Redshift-Datentypen und den Apache Iceberg-Tabellenspezifikationen.

Redshift-Datentyp	Aliasnamen	Iceberg-Datentyp
SMALLINT	INT2	int
INTEGER	GANZZAHL, INT4	int
BIGINT	INT8	long
DECIMAL	NUMERIC	Dezimalwert
REAL	FLOAT4	float
REAL	FLOAT4	float
DOUBLE PRECISION	FLOAT8, SCHWEBEN	double
CHAR	ZEICHEN, NCHAR	Zeichenfolge
VARCHAR	VARIIERENDER CHARAKTER, NVARCHAR	Zeichenfolge
BPCHAR		Zeichenfolge
TEXT		Zeichenfolge

Redshift-Datentyp	Aliasnamen	Iceberg-Datentyp
DATUM		date
TIME	ZEIT OHNE ZEITZONE	time
TIME	ZEIT MIT ZEITZONE	Nicht unterstützt
TIMESTAMP (ZEITSTEMPEL)	ZEITSTEMPEL OHNE ZEITZONE	TIMESTAMP (ZEITSTEMPEL)
ZEITSTEMPEL	ZEITSTEMPEL MIT ZEITZONE	ZEITSTEMPEL
INTERVAL YEAR TO MONTH		Nicht unterstützt
INTERVAL DAY TO SECOND		Nicht unterstützt
BOOLEAN	BOOL	bool
HLLSKETCH		Nicht unterstützt
SUPER		Nicht unterstützt
VARBYTE	VARBINARY, BINARY VARYING	Binary
GEOMETRY		Nicht unterstützt
GEOGRAPHY		Nicht unterstützt

Überlegungen und Einschränkungen bei der Verwendung von AWS Glue Iceberg REST Catalog APIs

Im Folgenden werden die Überlegungen und Einschränkungen bei der Verwendung des Betriebsverhaltens der Apache Iceberg REST Catalog Data Definition Language (DDL) beschrieben.

Überlegungen

- **`RenameTable`**API-Verhalten — Der `RenameTable` Vorgang wird in Tabellen in Amazon Redshift unterstützt, aber nicht in Amazon S3.
- DDL-Operationen für Namespaces und Tabellen in Amazon Redshift — Erstellungs-, Aktualisierungs- und Löschvorgänge für Namespaces und Tabellen in Amazon Redshift sind asynchrone Operationen, da sie davon abhängen, wann die von Amazon Redshift verwaltete Arbeitsgruppe verfügbar ist und ob eine widersprüchliche DDL- und DML-Transaktion läuft und der Vorgang auf die Sperre warten und dann versuchen muss, die Änderungen zu übernehmen.

Einschränkungen

- Ansichten APIs in der Apache Iceberg REST-Spezifikation werden im AWS Glue Iceberg REST Catalog nicht unterstützt.

AWS Glue Bewährte Methoden für den Datenkatalog

In diesem Abschnitt werden bewährte Methoden für die effektive Verwaltung und Nutzung von behandelt AWS Glue Data Catalog. Er konzentriert sich auf Praktiken wie effiziente Nutzung von Crawlern, Organisation von Metadaten, Sicherheit, Leistungsoptimierung, Automatisierung, Datenverwaltung und Integration mit anderen AWS Diensten.

- Effektiver Einsatz von Crawlern — Führen Sie regelmäßig Crawler aus, um den Datenkatalog up-to-date mit Änderungen in Ihren Datenquellen zu aktualisieren. Verwenden Sie inkrementelle Crawls für sich häufig ändernde Datenquellen, um die Leistung zu verbessern. Konfigurieren Sie Crawler so, dass sie automatisch neue Partitionen hinzufügen oder Schemas aktualisieren, wenn Änderungen erkannt werden.
- Organisieren und Benennen von Metadatentabellen — Richten Sie eine einheitliche Benennungskonvention für Datenbanken und Tabellen im Datenkatalog ein. Gruppieren Sie verwandte Datenquellen zur besseren Organisation in logische Datenbanken oder Ordner. Verwenden Sie aussagekräftige Namen, die den Zweck und den Inhalt der einzelnen Tabellen verdeutlichen.
- Effektives Verwalten von Schemas — Nutzen Sie die Schema-Inferenzfunktionen von Crawlern. AWS Glue Überprüfen und aktualisieren Sie Schemaänderungen, bevor Sie sie anwenden, um zu verhindern, dass nachgelagerte Anwendungen beschädigt werden. Verwenden Sie Funktionen zur Schemaentwicklung, um Schemaänderungen ordnungsgemäß zu handhaben.

- Sichern Sie den Datenkatalog — Aktivieren Sie die Datenverschlüsselung im Ruhezustand und bei der Übertragung für den Datenkatalog. Implementieren Sie differenzierte Richtlinien zur Zugriffskontrolle, um den Zugriff auf sensible Daten zu beschränken. Prüfen und überprüfen Sie regelmäßig die Berechtigungen und Aktivitätsprotokolle für den Datenkatalog.
- Integrieren Sie den Datenkatalog in andere AWS Dienste Verwenden Sie den Datenkatalog als zentrale Metadatenebene für Dienste wie Amazon Athena, Redshift Spectrum und. AWS Lake Formation Nutzen Sie AWS Glue ETL-Jobs, um Daten zu transformieren und in verschiedene Datenspeicher zu laden und gleichzeitig die Metadaten im Datenkatalog beizubehalten.
- Überwachen und optimieren Sie die Leistung Datenkatalog Überwachen Sie die Leistung von Crawlern und ETL-Jobs mithilfe von Amazon CloudWatch Metriken. Partitionieren Sie große Datensätze im Datenkatalog, um die Abfrageleistung zu verbessern. Implementieren Sie Leistungsoptimierungen für Metadaten, auf die häufig zugegriffen wird.
- Bleiben Sie mit der AWS Glue Dokumentation und den bewährten Methoden auf dem Laufenden. Datenkatalog Überprüfen Sie regelmäßig die AWS Glue Dokumentation und AWS Glue Ressourcen auf die neuesten Updates, bewährten Methoden und Empfehlungen. Nehmen Sie an AWS Glue Webinaren, Workshops und anderen Veranstaltungen teil, um von Experten zu lernen und über neue Funktionen und Möglichkeiten auf dem Laufenden zu bleiben.

Überwachung der Nutzungsmetriken des Datenkatalogs in Amazon CloudWatch

AWS Glue Data Catalog Nutzungsmetriken sind jetzt mit verfügbar Amazon CloudWatch, was die Überwachung und das Verständnis der Ressourcennutzung in Ihrem Datenkatalog vereinfacht. Sie haben jetzt sofort Einblick in die Nutzung von Katalogen, Datenbanken, Tabellen, Partitionen und Verbindungen durch Ihre Glue Catalog API, was es einfacher macht, den Überblick über Ihren Datenkatalog zu behalten.

Überblick über die Metriken von Data Catalog

AWS Glue Data Catalog veröffentlicht automatisch Nutzungsmetriken für Amazon CloudWatch. Mit der Integration von CloudWatch Metriken können Sie jede Minute wichtige Leistungsindikatoren verfolgen, darunter:

- Anfragen nach Tabellen
- Partitionsindizes wurden erstellt

- Verbindungen wurden aktualisiert
- Statistiken aktualisiert

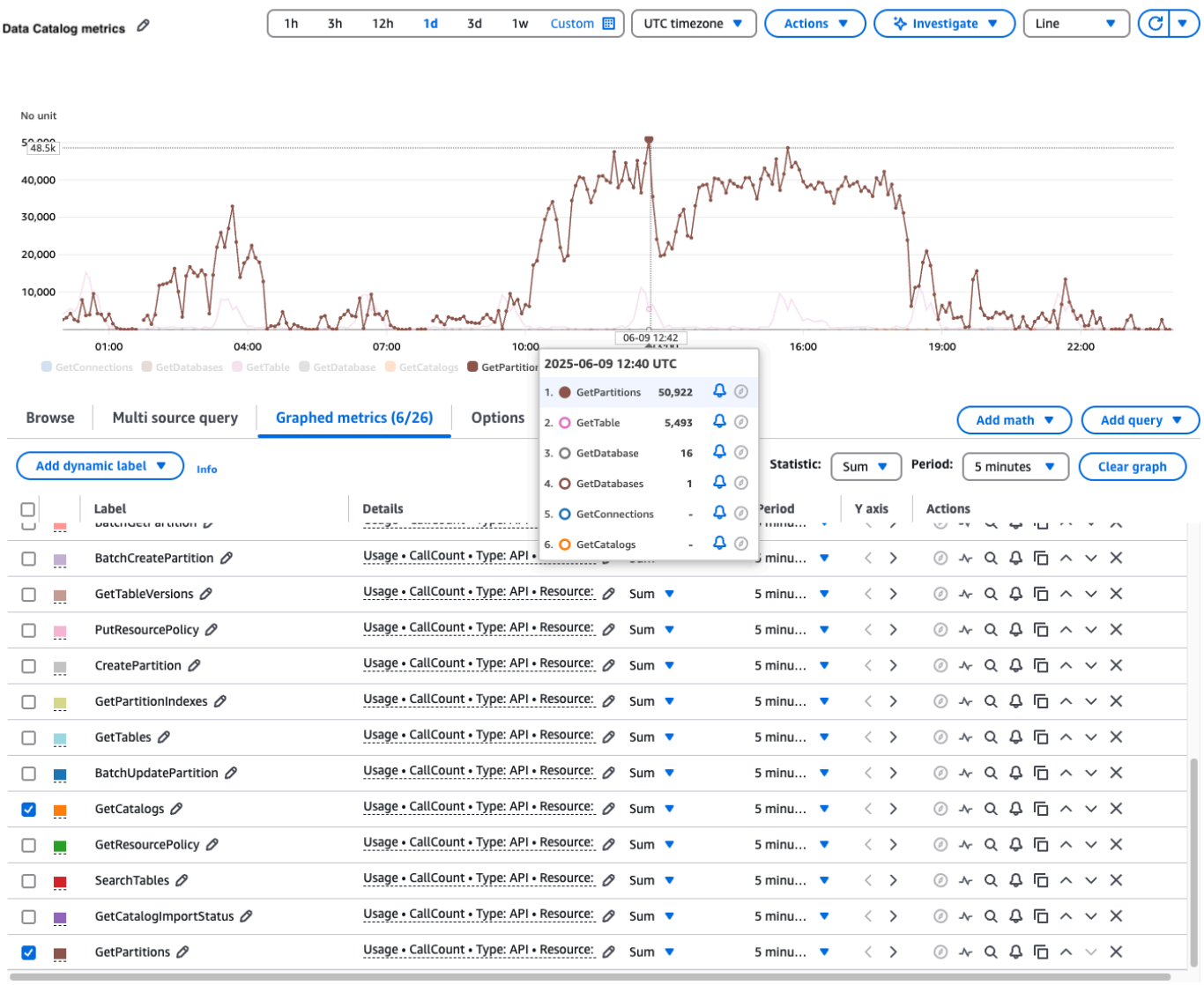
Diese Kennzahlen helfen Ihnen dabei, Engpässe zu identifizieren, Anomalien zu erkennen und datengestützte Entscheidungen zu treffen, um die allgemeine Zuverlässigkeit des Datenkatalogs zu verbessern. Sie können auch CloudWatch Alarme einrichten, um Benachrichtigungen zu erhalten, wenn Messwerte bestimmte Schwellenwerte überschreiten, sodass Sie Ihre Bereitstellung proaktiv verwalten können.

Hinzufügen von Metriken zu Ihrem Dashboard CloudWatch

Sie können benutzerdefinierte Dashboards erstellen, um Ihre AWS Glue Data Catalog Ressourcen zu überwachen, und Alarme einrichten, um über ungewöhnliche Aktivitäten informiert zu werden.

Gehen Sie wie folgt vor, um Datenkatalog-Metriken zu Ihrem CloudWatch Dashboard hinzuzufügen:

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie Alle Metriken aus.
4. Wählen Sie „Nutzung“ **>** „Nach Ressource AWS “.
5. Filtern Sie nach Glue, um die verfügbaren Metriken zu sehen.
6. Wählen Sie die Metriken aus, die Sie Ihrem Dashboard hinzufügen möchten.
7. Fügen Sie Ihrem CloudWatch Diagramm Metriken für Kataloge, Datenbanken, Tabellen, Partitionen und Verbindungen hinzu.



Sie können benutzerdefinierte Alarme konfigurieren, die automatisch ausgelöst werden, wenn die API-Nutzung Ihre definierten Schwellenwerte überschreitet, um Auffälligkeiten bei der Nutzung Ihres Datenkatalogs zu erkennen.

Detaillierte Anweisungen zum Einrichten von Alarmen finden Sie unter Einen Metrics CloudWatch Insights-Alarm erstellen.

AWS Glue Schema Registry

> **Note**
>
> AWS Glue Die Schemaregistrierung wird in den folgenden Regionen der AWS Glue Konsole nicht unterstützt: Asien-Pazifik (Jakarta), Europa (Zürich) und Naher Osten (VAE).

Das Tool AWS Glue Mit der Schemaregistrierung können Sie Datenstromschemas zentral ermitteln, steuern und weiterentwickeln. Ein Schema definiert die Struktur und das Format eines Datensatzes. Mit AWS Glue Mit der Schemaregistrierung können Sie Schemas in Ihren Datenstreaming-Anwendungen mithilfe praktischer Integrationen mit Apache Kafka, Amazon Kinesis Data Streams Amazon Managed Streaming for Apache Kafka, Amazon Managed Service for Apache Flink und verwalten und durchsetzen. AWS Lambda

Die Schemaregistry unterstützt das AVRO-Datenformat (v1.11.4), das JSON-Datenformat mit dem JSON-Schemaformat für das Schema (Spezifikationen Draft-04, Draft-06 und Draft-07) mit JSON-Schemavalidierung mithilfe der Everit-Bibliothek, die Protocol Buffers (Protobuf) -Versionen proto2 und proto3 ohne Unterstützung für `extensions` oder `groups` und die Java-Sprachunterstützung mit weiteren Datenformaten und Sprachen, die noch folgen werden. Zu den unterstützten Funktionen gehören Kompatibilität, Schemabeschaffung über Metadaten, automatische Registrierung von Schemas, IAM-Kompatibilität und optionale ZLIB-Komprimierung zur Reduzierung von Speicherplatz und Datenübertragung. Die Schema-Registry ist serverlos und kann kostenlos verwendet werden.

Die Verwendung eines Schemas als Datenformat-Vertrag zwischen Produzenten und Verbrauchern führt zu einer verbesserten Daten-Governance sowie einer höheren Datenqualität und ermöglicht Datenkonsumenten, resistent gegen kompatible Upstream-Änderungen zu sein.

Die Schemaregistrierung ermöglicht es unterschiedlichen Systemen, ein Schema für die Serialisierung und Deserialisierung gemeinsam zu nutzen. Angenommen Sie haben einen Produzenten und Verbraucher von Daten. Der Produzent kennt das Schema, wenn er die Daten veröffentlicht. Die Schema Registry stellt einen Serialisierer und einen Deserializer für bestimmte Systeme wie Amazon MSK oder Apache Kafka bereit.

Weitere Informationen finden Sie unter So funktioniert die Schemaregistrierung.

Themen

- Schemata

- Registrierungen
- Schema-Versioning und -Kompatibilität
- Open-Source-SerDe-Bibliotheken
- Kontingente in Schema Registry
- So funktioniert die Schemaregistrierung
- Erste Schritte mit der Schemaregistrierung
- Integration in AWS Glue Schema Registry
- Migration von einer Schemaregistrierung eines Drittanbieters zu AWS Glue Schema Registry

Schemata

Ein Schema definiert die Struktur und das Format eines Datensatzes. Ein Schema ist eine versionierte Spezifikation für zuverlässige Datenveröffentlichung, -nutzung oder -speicherung.

In diesem Beispielschema für Avro werden Format und Struktur durch die Layout- und Feldnamen definiert, und das Format der Feldnamen wird durch die Datentypen definiert (z. B.`string`, `int`).

```
{
    "type": "record",
    "namespace": "ABC_Organization",
    "name": "Employee",
    "fields": [
        {
            "name": "Name",
            "type": "string"
        },
        {
            "name": "Age",
            "type": "int"
        },
        {
            "name": "address",
            "type": {
                "type": "record",
                "name": "addressRecord",
                "fields": [
                    {
                        "name": "street",
                        "type": "string"
```

```
                    },
                    {
                        "name": "zipcode",
                        "type": "int"
                    }
                ]
            }
        }
    ]
}
```

In diesem Beispiel für das JSON-Schema draft-07 für JSON wird das Format von der JSON-Schemaorganisation definiert.

```
{
 "$id": "https://example.com/person.schema.json",
 "$schema": "http://json-schema.org/draft-07/schema#",
 "title": "Person",
 "type": "object",
 "properties": {
  "firstName": {
   "type": "string",
   "description": "The person's first name."
  },
  "lastName": {
   "type": "string",
   "description": "The person's last name."
  },
  "age": {
   "description": "Age in years which must be equal to or greater than zero.",
   "type": "integer",
   "minimum": 0
  }
 }
}
```

In diesem Beispiel für Protobuf wird das Format durch Version 2 der Protocol Buffers Sprache (proto2) definiert.

```
syntax = "proto2";

package tutorial;
```

```
option java_multiple_files = true;
option java_package = "com.example.tutorial.protos";
option java_outer_classname = "AddressBookProtos";

message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    optional string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }

  repeated PhoneNumber phones = 4;
}

message AddressBook {
  repeated Person people = 1;
}
```

Registrierungen

Eine Registry ist ein logischer Container mit Schemata. Eine Registry ermöglicht Ihnen, Ihre Schemata zu organisieren und die Zugriffskontrolle für Ihre Anwendungen zu verwalten.
Eine Registry verfügt über einen Amazon Resource Name (ARN), mit dem Sie verschiedene Zugriffsberechtigungen für Schemavorgänge in der Registry organisieren und festlegen können.

Sie können die Standard-Registry verwenden oder so viele neue Registries erstellen, wie nötig.

AWS Glue Hierarchie der Schemaregistrierung

- RegistryName: [Zeichenfolge]
  - RegistryArn: [AWS ARN]
  - CreatedTime: [Zeitstempel]

- UpdatedTime: [Zeitstempel]

- SchemaName: [Zeichenfolge]
  - SchemaArn: [AWS ARN]
  - DataFormat: [Avro, Json oder Protobuf]
  - Kompatibilität: [z. B. ABWÄRTS, ABWÄRTS_ALLE, AUFWÄRTS, AUFWÄRTS_ALLE, VOLL, VOLL_ALLE, KEINE, DEAKTIVIERT]
  - Status: [z. B. AUSSTEHEND, VERFÜGBAR, LÖSCHEN]
  - SchemaCheckpoint: [Ganzzahl]
  - CreatedTime: [Zeitstempel]
  - UpdatedTime: [Zeitstempel]
- SchemaVersion: [Zeichenfolge]
  - SchemaVersionNumber: [Ganzzahl]
  - Status: [z. B. AUSSTEHEND, VERFÜGBAR, LÖSCHEN, FEHLER]
  - SchemaDefinition: [Zeichenfolge, Wert: JSON]
  - CreatedTime: [Zeitstempel]
- SchemaVersionMetadata: [Liste]
  - MetadataKey: [Zeichenfolge]
  - MetadataInfo
  - MetadataValue: [Zeichenfolge]
  - CreatedTime: [Zeitstempel]

Schema-Versioning und -Kompatibilität

Jedes Schema kann mehrere Versionen haben. Das Versioning wird durch eine Kompatibilitätsregel geregelt, die auf ein Schema angewendet wird. Anforderungen zur Registrierung neuer Schemaversionen werden von der Schema Registry anhand dieser Regel geprüft, bevor sie durchgeführt werden.

Eine Schemaversion, die als Checkpoint markiert ist, wird verwendet, um die Kompatibilität der Registrierung neuer Versionen eines Schemas zu ermitteln. Wenn ein Schema zum ersten Mal

erstellt wird, ist der Standard-Checkpoint die erste Version. Wenn sich das Schema mit weiteren Versionen entwickelt, können Sie CLI/SDK verwenden, um den Checkpoint auf eine Version eines Schemas zu ändern, indem Sie die `UpdateSchema`-API verwenden, die eine Reihe von Einschränkungen einhält. Wenn Sie die Schemadefinition oder den Kompatibilitätsmodus in der Konsole bearbeiten, wird der Checkpoint standardmäßig auf die neueste Version geändert.

Mit Kompatibilitätsmodi können Sie steuern, wie sich Schemata im Lauf der Zeit entwickeln können oder nicht. Diese Modi bilden den Vertrag zwischen Anwendungen, die Daten erzeugen und Anwendungen, die Daten verbrauchen. Wenn eine neue Version eines Schemas an die Registrierung übermittelt wird, wird anhand der auf den Schemanamen angewendeten Kompatibilitätsregel ermittelt, ob die neue Version akzeptiert werden kann. Es gibt 8 Kompatibilitätsmodi: NONE (KEINE), DISABLED (DEAKTIVIERT), BACKWARD (ABWÄRTS), BACKWARD_ALL (ABWÄRTS_ALLE), FORWARD (AUFWÄRTS), FORWARD_ALL (AUFWÄRTS_ALLE), FULL (VOLL), FULL_ALL (VOLL_ALLE).

Im Avro-Datenformat können Felder optional oder erforderlich sein. Ein optionales Feld ist ein Feld, in dem `Type` null enthält. Erforderliche Felder haben keine NULL-Werte als `Type`.

Im Protobuf-Datenformat können Felder in der Proto2-Syntax optional (einschließlich wiederholt) oder erforderlich sein, während in der Proto3-Syntax alle Felder optional (einschließlich wiederholt) sind. Alle Kompatibilitätsregeln werden auf der Grundlage des Verständnisses der Protokollpufferspezifikationen sowie der Leitlinien der Dokumentation zu Google Protocol Buffers bestimmt.

- NONE (KEINE): Es gibt keinen Kompatibilitätsmodus. Sie können diese Option in Entwicklungsszenarien verwenden, oder wenn Sie die Kompatibilitätsmodi, die Sie auf Schemata anwenden möchten, nicht kennen. Jede neue Version, die hinzugefügt wird, wird ohne Kompatibilitätsprüfung akzeptiert.
- DISABLED (DEAKTIVIERT): Diese Kompatibilitätsoption verhindert das Versioning für ein bestimmtes Schema. Es können keine neuen Versionen hinzugefügt werden.
- BACKWARD (ABWÄRTS): Diese Kompatibilitätsoption wird empfohlen, damit Verbraucher sowohl die aktuelle als auch die vorherige Schemaversion lesen können. Sie können diese Option verwenden, wenn Sie Felder löschen oder optionale Felder hinzufügen und die Kompatibilität mit der vorherigen Schemaversion überprüfen möchten. Ein typischer Anwendungsfall für BACKWARD (ABWÄRTS) ist, wenn Ihre Anwendung für das neueste Schema erstellt wurde.

  AVRO

Angenommen Sie haben ein Schema mit Vorname (erforderlich), Nachname (erforderlich), E-Mail (erforderlich) und Telefonnummer (optional) erstellt.

Wenn Ihre nächste Schemaversion das erforderliche E-Mail-Feld entfernt, würde dies erfolgreich registriert werden. Abwärtskompatibilität erfordert, dass Verbraucher die aktuelle und die vorherige Schemaversion lesen können. Ihre Verbraucher können das neue Schema lesen, da das zusätzliche E-Mail-Feld aus alten Nachrichten ignoriert wird.

Wenn Sie eine vorgeschlagene neue Schemaversion haben, die ein erforderliches Feld hinzufügt, z. B. Postleitzahl, würde dies bei einer erforderlichen Abwärtskombatibilität nicht erfolgreich registriert. Verbraucher der neuen Version könnten alte Nachrichten vor der Schemaänderung nicht lesen, da ihnen das erforderliche PLZ-Feld fehlt. Wenn das PLZ-Feld jedoch im neuen Schema als optional festgelegt wurde, würde sich die vorgeschlagene Version erfolgreich registrieren, da Verbraucher das alte Schema ohne das optionale PLZ-Feld lesen können.

JSON

Angenommen Sie haben eine Schemaversion mit Vorname (optional), Nachname (optional), E-Mail (optional) und Telefonnummer (optional).

Wenn Ihre nächste Schemaversion die optionale Telefonnummereigenschaft hinzufügt, würde dies erfolgreich registriert werden, solange die ursprüngliche Schemaversion keine zusätzlichen Eigenschaften zulässt und das `additionalProperties`-Feld auf „False“ festlegt. Abwärtskompatibilität erfordert, dass Verbraucher die aktuelle und die vorherige Schemaversion lesen können. Ihre Verbraucher können Daten lesen, die mit dem ursprünglichen Schema erstellt wurden, in dem die Telefonnummereigenschaft nicht existiert.

Wenn Sie eine vorgeschlagene neue Schemaversion haben, die die optionale Telefonnummereigenschaft hinzufügt, würde dies mit der Abwärtskompatibilität nicht erfolgreich registriert, wenn die ursprüngliche Schemaversion das `additionalProperties`-Feld auf „True“ setzt und eine zusätzliche Eigenschaft erlaubt. Ihre Verbraucher mit der neuen Version könnten alte Nachrichten vor der Schemaänderung nicht lesen, da sie keine Daten mit der Telefonnummereigenschaft als anderen Typ lesen können, z. B. „Zeichenfolge“ statt „Ganzzahl“.

PROTOBUF

Angenommen, Sie haben ein Schema mit einer Nachricht `Person` mit den Feldern `first name` (erforderlich), `last name` (erforderlich), `email` (erforderlich) und `phone number` (optional) unter proto2-Syntax definiert.

Ähnlich wie bei AVRO-Szenarien, wenn Ihre nächste Schemaversion das erforderliche `email`-Feld entfernt, würde dies erfolgreich registriert werden. Abwärtskompatibilität erfordert, dass Verbraucher die aktuelle und die vorherige Schemaversion lesen können. Ihre Verbraucher können das neue Schema lesen, da das zusätzliche `email`-Feld aus alten Nachrichten ignoriert wird.

Wenn Sie eine vorgeschlagene neue Schemaversion haben, die ein erforderliches Feld hinzufügt, z. B. `zip code`, würde dies bei einer erforderlichen Abwärtskombatibilität nicht erfolgreich registriert. Verbraucher der neuen Version könnten alte Nachrichten vor der Schemaänderung nicht lesen, da ihnen das erforderliche `zip code`-Feld fehlt. Wenn das `zip code`-Feld jedoch im neuen Schema als optional festgelegt wurde, würde sich die vorgeschlagene Version erfolgreich registrieren, da Verbraucher das alte Schema ohne das optionale `zip code`-Feld lesen können.

Im Falle eines gRPC-Anwendungsfalls ist das Hinzufügen eines neuen RPC-Services oder einer RPC-Methode eine abwärtskompatible Änderung. Angenommen, Sie haben ein Schema mit einem RPC-Service `MyService` mit zwei RPC-Methoden `Foo` und `Bar` definiert.

Wenn Ihre nächste Schemaversion eine neue RPC-Methode namens `Baz` hinzufügt, würde dies erfolgreich registriert werden. Ihre Verbraucher können Daten lesen, die mit dem ursprünglichen Schema erstellt wurden, entsprechend der ABWÄRTSkompatibilität, da die neu hinzugefügte RPC-Methode `Baz` optional ist.

Wenn Sie eine neue vorgeschlagene Schemaversion haben, die die bestehende RPC-Methode `Foo` entfernt, würde diese nicht erfolgreich mit der RÜCKWÄRTSkompatibilität registriert. Ihre Verbraucher mit der neuen Version könnten alte Nachrichten vor der Schemaänderung nicht lesen, da sie Daten mit der nicht vorhandenen RPC-Methode `Foo` in einer gRPC-Anwendung nicht verstehen und lesen können.

- BACKWARD_ALL (ABWÄRTS_ALLE): Diese Kompatibilitätsoption ermöglicht Verbrauchern, sowohl die aktuelle als auch alle vorherigen Schemaversionen zu lesen. Sie können diese Option verwenden, wenn Sie Felder löschen oder optionale Felder hinzufügen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.
- FORWARD (AUFWÄRTS): Diese Kompatibilitätsoption ermöglicht Verbrauchern, sowohl die aktuelle als auch die nachfolgende Schemaversion zu lesen, aber nicht unbedingt spätere Versionen. Sie können diese Option verwenden, wenn Sie Felder hinzufügen oder optionale

Felder löschen und die Kompatibilität mit der vorherigen Schemaversion überprüfen möchten. Ein typischer Anwendungsfall für FORWARD (AUFWÄRTS) ist, wenn Ihre Anwendung für ein vorheriges Schema erstellt wurde und in der Lage sein soll, ein neueres Schema zu verarbeiten.

AVRO

Angenommen Sie haben ein Schema mit Vorname (erforderlich), Nachname (erforderlich) und E-Mail (optional) erstellt.

Wenn Sie eine neue Schemaversion haben, die ein erforderliches Feld hinzufügt, z. B. Telefonnummer, würde dies erfolgreich registriert werden. Aufwärtskompatibilität erfordert, dass Verbraucher Daten lesen können, die mit dem neuen Schema erstellt wurden, indem sie die vorherige Version verwenden.

Wenn Sie eine vorgeschlagene Schemaversion haben, die das erforderliche Vornamenfeld löscht, würde dies bei einer Aufwärtskompatibilität nicht erfolgreich registriert. Verbraucher mit der vorherigen Version könnten die vorgeschlagenen Schemata nicht lesen, da ihnen das erforderliche Vornamenfeld fehlt. Wenn das Feld für den Vornamen ursprünglich optional war, würde das vorgeschlagene neue Schema erfolgreich registriert werden, da die Verbraucher Daten basierend auf dem neuen Schema lesen können, die nicht über das optionale Feld Vorname verfügen.

JSON

Angenommen Sie haben eine Schemaversion mit Vorname (optional), Nachname (optional), E-Mail (optional) und Telefonnummer (optional).

Wenn Sie eine neue Schemaversion haben, die die optionale Telefonnummereigenschaft entfernt, würde dies erfolgreich registriert werden, solange die neue Schemaversion keine zusätzlichen Eigenschaften zulässt und das `additionalProperties`-Feld auf „False“ festlegt. Aufwärtskompatibilität erfordert, dass Verbraucher Daten lesen können, die mit dem neuen Schema erstellt wurden, indem sie die vorherige Version verwenden.

Wenn Sie eine vorgeschlagene Schemaversion haben, die die optionale Telefonnummereigenschaft entfernt, würde dies mit der Aufwärtskompatibilität nicht erfolgreich registriert, wenn die neue Schemaversion das `additionalProperties`-Feld auf „True“ setzt und eine zusätzliche Eigenschaft erlaubt. Ihre Verbraucher mit der vorherigen Version könnten die vorgeschlagenen Schemata nicht lesen, da diese die Telefonnummereigenschaft als anderen Typ haben könnten, z. B. „Zeichenfolge“ statt „Ganzzahl“.

PROTOBUF

Angenommen, Sie haben ein Schema mit einer Nachricht `Person` mit den Feldern `first name` (erforderlich), `last name` (erforderlich) und `email` (optional) unter proto2-Syntax definiert.

Ähnlich wie in den AVRO-Szenarien, wenn Sie eine neue Schemaversion haben, die ein erforderliches Feld hinzufügt, z. B. `phone number`, würde dies erfolgreich registriert werden. Aufwärtskompatibilität erfordert, dass Verbraucher Daten lesen können, die mit dem neuen Schema erstellt wurden, indem sie die vorherige Version verwenden.

Wenn Sie eine vorgeschlagene Schemaversion haben, die das erforderliche Feld `first name` löscht, würde dies bei einer VORWÄRTSkompatibilität nicht erfolgreich registriert. Verbraucher mit der vorherigen Version könnten die vorgeschlagenen Schemata nicht lesen, da ihnen das erforderliche Feld `first name` fehlt. Wenn das Feld `first name` ursprünglich optional war, würde das vorgeschlagene neue Schema erfolgreich registriert werden, da die Verbraucher Daten basierend auf dem neuen Schema lesen können, die nicht über das optionale Feld `first name` verfügen.

Im Falle eines gRPC-Anwendungsfalls ist das Entfernen eines RPC-Services oder einer RPC-Methode eine vorwärtskompatible Änderung. Angenommen, Sie haben ein Schema mit einem RPC-Service `MyService` mit zwei RPC-Methoden `Foo` und `Bar` definiert.

Wenn Ihre nächste Schemaversion die bestehende RPC-Methode mit dem Namen `Foo` löscht, würde dies gemäß der VORWÄRTSkompatibilität erfolgreich registriert, da die Verbraucher Daten, die mit dem neuen Schema erzeugt wurden, unter Verwendung der vorherigen Version lesen können. Wenn Sie eine vorgeschlagene neue Schemaversion haben, die eine RPC-Methode `Baz` hinzufügt, würde diese mit der VORWÄRTSkompatibilität nicht erfolgreich registriert. Verbraucher mit der vorherigen Version könnten die vorgeschlagenen Schemata nicht lesen, da ihnen die erforderliche RPC-Methode `Baz` fehlt.

- FORWARD_ALL (AUFWÄRTS_ALLE): Diese Kompatibilitätsoption ermöglicht Verbrauchern, Daten zu lesen, die von Verfassern eines neuen registrierten Schemas geschrieben wurden. Sie können diese Option verwenden, wenn Sie Felder hinzufügen oder optionale Felder löschen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.
- FULL (VOLL): Diese Kompatibilitätsoption ermöglicht Verbrauchern, Daten zu lesen, die von Verfassern geschrieben wurden, die die vorherige oder die nächste Version des Schemas verwenden, jedoch nicht frühere oder spätere Versionen. Sie können diese Option verwenden,

wenn Sie optionale Felder hinzufügen oder löschen und die Kompatibilität mit der vorherigen Schemaversion überprüfen möchten.

- FULL_ALL (VOLL_ALLE): Diese Kompatibilitätsoption ermöglicht Verbrauchern, Daten zu lesen, die von Verfassern geschrieben wurden, die alle früheren Schemaversionen verwenden. Sie können diese Option verwenden, wenn Sie optionale Felder hinzufügen oder löschen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.

Open-Source-SerDe-Bibliotheken

AWS bietet Open-Source-Serde-Bibliotheken als Framework für die Serialisierung und Deserialisierung von Daten. Das Open-Source-Design dieser Bibliotheken ermöglicht gängige Open-Source-Anwendungen und Frameworks, diese Bibliotheken in ihren Projekten zu unterstützen.

Weitere Details zur Funktionsweise der SerDe-Bibliotheken finden Sie unter So funktioniert die Schemaregistrierung.

Kontingente in Schema Registry

Kontingente, auch Limits genannt AWS, sind die Höchstwerte für die Ressourcen, Aktionen und Elemente in Ihrem Konto. AWS Im Folgenden finden Sie weiche Grenzwerte für die Schemaregistrierung in AWS Glue.

Metadaten-Schlüssel-Wert-Paare pro Schemaversion.

Sie können bis zu 10 Schlüssel-Wert-Paare SchemaVersion pro Region verwenden. AWS

Sie können die Schlüssel-Wert-Metadatenpaare mit dem oder anzeigen oder festlegen. QuerySchemaVersionMetadata Aktion (Python: query_schema_version_metadata) PutSchemaVersionMetadata Aktion (Python: put_schema_version_metadata) APIs

Im Folgenden finden Sie feste Grenzwerte für die Schemaregistrierung in AWS Glue.

Registrierungen

Sie können bis zu 100 Registrierungen pro AWS Region für dieses Konto einrichten.

SchemaVersion

Sie können bis zu 10000 Schemaversionen pro AWS Region für dieses Konto haben.

Jedes neue Schema erstellt eine neue Schemaversion, sodass Sie theoretisch bis zu 10000 Schemas pro Konto und Region haben können, wenn jedes Schema nur eine Version hat.

Schema-Nutzlasten

Es gibt eine Größenbeschränkung von 170 KB für Schema-Nutzlasten.

So funktioniert die Schemaregistrierung

In diesem Abschnitt wird beschrieben, wie die Serialisierungs- und Deserialisierungsprozesse in der Schemaregistrierung funktionieren.

1. Registrieren eines Schemas: Wenn das Schema noch nicht in der Registrierung vorhanden ist, kann das Schema mit einem Schemanamen registriert werden, der dem Namen des Ziels entspricht (z. B. test_topic, test_stream, prod_firehose) oder der Produzent kann einen benutzerdefinierten Namen für das Schema angeben. Produzenten können dem Schema auch Schlüssel-Wert-Paare als Metadaten hinzufügen, z. B. Quelle: MSK_Kafka_Topic_A, oder bei der Schemaerstellung Tags auf Schemas anwenden. AWS Sobald ein Schema registriert ist, gibt Schema Registry die Schemaversions-ID an den Serializer zurück. Wenn das Schema vorhanden ist, aber der Serializer eine neue Version verwendet, die nicht existiert, überprüft Schema Registry die Schemareferenz, eine Kompatibilitätsregel, um sicherzustellen, dass die neue Version kompatibel ist, bevor sie als neue Version registriert wird.

   Es gibt zwei Methoden zum Registrieren eines Schemas: manuelle Registrierung und automatische Registrierung. Sie können ein Schema manuell registrieren über AWS Glue Konsole oder CLI/SDK.

   Wenn die automatische Registrierung in den Einstellungen des Serializers aktiviert ist, wird die automatische Registrierung des Schemas durchgeführt. Wenn `REGISTRY_NAME` nicht in den Konfigurationen des Produzenten bereitgestellt wird, registriert die automatische Registrierung die neue Schemaversion in der Standardregistrierung (Default-Registry). Siehe SerDe Bibliotheken installieren für Informationen zur Angabe der Eigenschaft für die automatische Registrierung.
2. Der Serializer validiert Datensätze anhand des Schemas: Wenn die Anwendung, die Daten erzeugt, ihr Schema registriert hat, überprüft der Serializer der Schema Registry, ob der Datensatz, der von der Anwendung erstellt wird, mit den Feldern und Datentypen strukturiert ist, die einem registrierten Schema entsprechen. Wenn das Schema des Datensatzes nicht mit einem registrierten Schema übereinstimmt, gibt der Serializer eine Ausnahme zurück und die Anwendung kann den Datensatz nicht an das Ziel liefern.

Wenn kein Schema vorhanden ist und der Schemaname nicht über die Konfigurationen des Produzenten bereitgestellt wird, wird das Schema mit demselben Namen wie der Themenname (bei Apache Kafka oder Amazon MSK) oder der Streamname (bei Kinesis Data Streams) erstellt.

Jeder Datensatz hat eine Schemadefinition und Daten. Die Schemadefinition wird anhand der vorhandenen Schemata und Versionen in der Schema Registry abgefragt.

Standardmäßig speichern die Produzenten Schemadefinitionen und die Schemaversion IDs registrierter Schemas im Cache. Wenn die Schemaversionsdefinition eines Datensatzes nicht mit dem übereinstimmt, was im Cache verfügbar ist, versucht der Produzent, das Schema mit der Schema Registry zu validieren. Wenn die Schemaversion gültig ist, werden ihre Versions-ID und Definition lokal beim Produzenten zwischengespeichert.

Sie können den Standard-Cache-Zeitraum (24 Stunden) in den optionalen Producer-Eigenschaften in Schritt 3 von SerDe Bibliotheken installieren anpassen.

3. Serialisieren und Ausliefern von Datensätzen: Wenn der Datensatz dem Schema entspricht, kennzeichnet der Serializer jeden Datensatz mit der Schema-Versions-ID, serialisiert den Datensatz basierend auf dem ausgewählten Datenformat (AVRO, JSON, Protobuf oder in Kürze weitere Formate), komprimiert den Datensatz (optionale Konfiguration des Produzenten) und liefert ihn an das Ziel.
4. Verbraucher deserialisieren die Daten: Verbraucher, die diese Daten lesen, verwenden die Deserializer-Bibliothek der Schema Registry, die die Schemaversions-ID aus der Datensatznutzlast analysiert.
5. Der Deserializer kann das Schema von der Schema Registry anfordern: Wenn dies das erste Mal ist, dass der Deserializer Datensätze mit einer bestimmten Schemaversions-ID sieht, fordert der Deserializer das Schema unter Verwendung der Schemaversions-ID von der Schema-Registry an und speichert das Schema lokal auf dem Verbraucher zwischen. Wenn die Schema Registry den Datensatz nicht deserialisieren kann, kann der Verbraucher die Daten aus dem Datensatz protokollieren und fortfahren oder die Anwendung anhalten.
6. Der Deserializer verwendet das Schema, um den Datensatz zu deserialisieren: Wenn der Deserializer die Schemaversions-ID aus der Schema Registry abruft, dekomprimiert der Deserializer den Datensatz (wenn der vom Produzent gesendete Datensatz komprimiert wird) und verwendet das Schema, um den Datensatz zu deserialisieren. Dann verarbeitet die Anwendung den Datensatz.

> **Note**
>
> Verschlüsselung: Ihre Clients kommunizieren mit der Schema Registry über API-Aufrufe, die Daten während der Übertragung mittels TLS-Verschlüsselung über HTTPS verschlüsseln. In der Schemaregistry gespeicherte Schemas werden im Ruhezustand immer mit einem vom Dienst verwalteten Schlüssel AWS Key Management Service ()AWS KMS verschlüsselt.

> **Note**
>
> Benutzerautorisierung: Die Schema Registry unterstützt identitätsbasierte IAM-Richtlinien.

Erste Schritte mit der Schemaregistrierung

Die folgenden Abschnitte vermitteln einen Überblick und führen Sie durch das Einrichten und Verwenden der Schema Registry. Informationen zu Konzepten und Komponenten der Schemaregistrierung finden Sie unterAWS Glue Schema Registry.

Themen

- SerDe Bibliotheken installieren
- Erstellen einer Registrierung
- Erstellen eines Schemas
- Aktualisieren eines Schemas oder einer Registrierung
- Ein Schema oder eine Registrierung löschen
- IAM-Beispiele für Serialisierer
- IAM-Beispiele für Deserialisierer
- Private Konnektivität mit AWS PrivateLink
- Zugreifen auf CloudWatch Amazon-Metriken
- AWS CloudFormation Beispielvorlage für die Schemaregistrierung

SerDe Bibliotheken installieren

> **Note**
>
> Bevor Sie die folgenden Schritte ausführen, müssen Sie einen aktiven Amazon-MSK- (Amazon Managed Streaming for Apache Kafka) oder Apache-Kafka-Cluster haben. Ihre Produzenten und Verbraucher müssen Java 8 oder höher nutzen.

Die SerDe Bibliotheken bieten ein Framework für die Serialisierung und Deserialisierung von Daten.

Sie installieren den Open-Source-Serializer für Ihre Anwendungen, die Daten erzeugen (zusammen die „Serializer“). Der Serializer übernimmt Serialisierung, Komprimierung und Interaktion mit der Schema Registry. Der Serializer extrahiert das Schema automatisch aus einem Datensatz, der in ein mit der Schema Registry kompatibles Ziel wie Amazon MSK geschrieben wird. Ebenso installieren Sie den Open-Source-Deserializer auf Ihren Anwendungen, die Daten verbrauchen.

So installieren Sie die Bibliotheken auf Produzenten und Verbrauchern:

1. Fügen Sie diese Abhängigkeit in den pom.xml-Dateien von Produzenten und Verbrauchern über den folgenden Code hinzu:

   ```
   <dependency>
       <groupId>software.amazon.glue</groupId>
       <artifactId>schema-registry-serde</artifactId>
       <version>1.1.5</version>
   </dependency>
   ```

   Alternativ können Sie das klonen AWS Glue Github-Repository der Schemaregistrierung.

2. Richten Sie Ihre Produzenten mit den folgenden Eigenschaften ein:

   ```
   props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
    StringSerializer.class.getName()); // Can replace StringSerializer.class.getName())
    with any other key serializer that you may use
   props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
    GlueSchemaRegistryKafkaSerializer.class.getName());
   props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
   properties.put(AWSSchemaRegistryConstants.DATA_FORMAT, "JSON"); // OR "AVRO"
   ```

Wenn keine Schemas vorhanden sind, muss die automatische Registrierung aktiviert werden (nächster Schritt). Wenn Sie ein Schema haben, das Sie anwenden möchten, ersetzen Sie „my-schema“ durch Ihren Schemanamen. Auch der „registry-name“ muss angegeben werden, wenn die automatische Registrierung des Schemas deaktiviert ist. Wenn das Schema unter der „default-registry“ erstellt wird, kann der Registrierungsname weggelassen werden.

3. (Optional) Legen Sie eine dieser optionalen Produzenteneigenschaften fest. Detaillierte Eigenschaftsbeschreibungen finden Sie in der ReadMe Datei.

```
props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, "true"); // If
 not passed, uses "false"
props.put(AWSSchemaRegistryConstants.SCHEMA_NAME, "my-schema"); // If not passed,
 uses transport name (topic name in case of Kafka, or stream name in case of Kinesis
 Data Streams)
props.put(AWSSchemaRegistryConstants.REGISTRY_NAME, "my-registry"); // If not passed,
 uses "default-registry"
props.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); // If
 not passed, uses 86400000 (24 Hours)
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.COMPATIBILITY_SETTING, Compatibility.FULL); //
 Pass a compatibility mode. If not passed, uses Compatibility.BACKWARD
props.put(AWSSchemaRegistryConstants.DESCRIPTION, "This registry is used for several
 purposes."); // If not passed, constructs a description
props.put(AWSSchemaRegistryConstants.COMPRESSION_TYPE,
 AWSSchemaRegistryConstants.COMPRESSION.ZLIB); // If not passed, records are sent
 uncompressed
```

Die automatische Registrierung registriert die Schemaversion unter der Standardregistrierung („default-registry“). Wenn im vorherigen Schritt kein `SCHEMA_NAME` angegeben wird, wird der Themenname als `SCHEMA_NAME` verwendet.

Weitere Informationen zu Kompatibilitätsmodi finden Sie unter Schema-Versioning und -Kompatibilität.

4. Richten Sie Ihre Verbraucher mit den folgenden Eigenschaften ein:

```
props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 GlueSchemaRegistryKafkaDeserializer.class.getName());
props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2"); // Pass an AWS-Region
```

```
props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName()); // Only required for AVRO data format
```

5. (Optional) Legen Sie diese optionalen Verbrauchereigenschaften fest. Detaillierte Eigenschaftsbeschreibungen finden Sie in der ReadMe Datei.

```
properties.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); //
 If not passed, uses 86400000
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 "com.amazonaws.services.schemaregistry.deserializers.external.ThirdPartyDeserializer"); //
 For migration fall back scenario
```

Erstellen einer Registrierung

Sie können die Standardregistrierung verwenden oder so viele neue Registrierungen wie nötig erstellen, indem Sie AWS Glue APIs oder AWS Glue console.

AWS Glue APIs

Sie können diese Schritte verwenden, um diese Aufgabe mit dem AWS Glue APIs.

Um das zu verwenden AWS CLI für AWS Glue Stellen Sie sicher APIs, dass Sie Ihre Schemaregistrierung AWS CLI auf die neueste Version aktualisieren.

Um eine neue Registrierung hinzuzufügen, verwenden Sie die CreateRegistry Aktion (Python: create_registry)-API. Geben Sie `RegistryName` als Name der zu erstellenden Registrierung an (maximale Länge 255 Zeichen, nur Buchstaben, Zahlen, Bindestriche, Unterstriche, Dollarzeichen oder Hashzeichen).

Geben Sie a `Description` als Zeichenfolge an, die nicht mehr als 2048 Byte lang ist und dem mehrzeiligen Zeichenkettenmuster der URI-Adresse entspricht.

Geben Sie optional einen oder mehrere `Tags` als Map-Array für Schlüssel-Wert-Paare Ihrer Registrierung an.

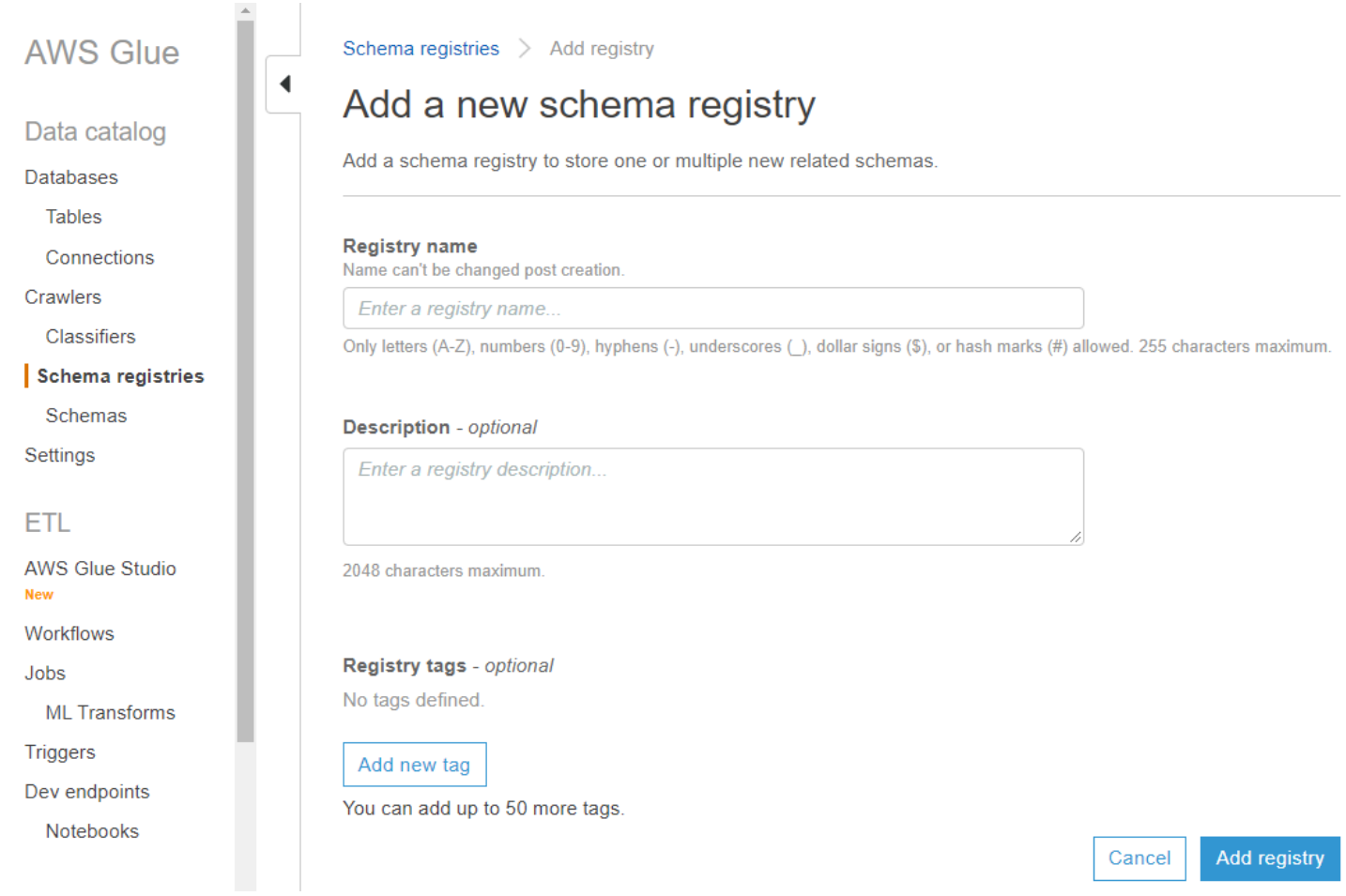
```
aws glue create-registry --registry-name registryName1 --description description
```

Wenn Ihre Registrierung erstellt wurde, wird ihr ein Amazon-Ressourcenname (ARN) zugewiesen, den Sie im Abschnitt `RegistryArn` der API-Antwort anzeigen können. Nachdem Sie nun eine Registrierung erstellt haben, erstellen Sie ein oder mehrere Schemata für diese Registrierung.

AWS Glue Konsole

Um eine neue Registrierung in der AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schema Registries (Schemaregistrierungen).
3. Wählen Sie Add registry (Registrierung hinzufügen).
4. Geben Sie einen Registry name (Registrierungsname) aus Buchstaben, Ziffern, Bindestrichen oder Unterstrichen ein. Dieser Name kann nicht geändert werden.
5. Geben Sie eine Description (Beschreibung) (optional) für die Registrierung ein.
6. Wenden Sie optional einen oder mehrere Tags auf Ihre Registrierung an. Klicken Sie auf Add new Tag (Neuen Tag hinzufügen) und geben Sie einen Tag-Schlüssel und optional einen Tag-Wert ein.
7. Wählen Sie Add registry (Registrierung hinzufügen).



Wenn Ihre Registrierung erstellt wird, wird ihr ein Amazon-Ressourcenname (ARN) zugewiesen, den Sie anzeigen können, indem Sie die Registrierung aus der Liste in Schema Registries (Schemaregistrierungen) auswählen. Nachdem Sie nun eine Registrierung erstellt haben, erstellen Sie ein oder mehrere Schemata für diese Registrierung.

Umgang mit einem bestimmten Datensatz (JAVA POJO) für JSON

Sie können ein Plain-Old-Java-Objekt (POJO) verwenden und das Objekt als Datensatz übergeben. Dies ist ähnlich dem Konzept eines bestimmten Datensatzes in AVRO. Das mbknor-jackson-jsonschemakann ein JSON-Schema für das übergebene POJO generieren. Diese Bibliothek kann auch zusätzliche Informationen in das JSON-Schema einfügen.

Das Tool AWS Glue Die Schemaregistrierungsbibliothek verwendet das eingefügte Feld „ClassName“ im Schema, um einen vollständig klassifizierten Klassennamen bereitzustellen. Das Feld „className“ wird vom Deserializer verwendet, um in ein Objekt dieser Klasse zu deserialisieren.

```
 Example class :

@JsonSchemaDescription("This is a car")
@JsonSchemaTitle("Simple Car Schema")
@Builder
@AllArgsConstructor
@EqualsAndHashCode
// Fully qualified class name to be added to an additionally injected property
// called className for deserializer to determine which class to deserialize
// the bytes into
@JsonSchemaInject(
        strings = {@JsonSchemaString(path = "className",
                value =
 "com.amazonaws.services.schemaregistry.integrationtests.generators.Car")}
)
// List of annotations to help infer JSON Schema are defined by https://github.com/
mbknor/mbknor-jackson-jsonSchema
public class Car {
    @JsonProperty(required = true)
    private String make;

    @JsonProperty(required = true)
    private String model;

    @JsonSchemaDefault("true")
    @JsonProperty
```

```
    public boolean used;

    @JsonSchemaInject(ints = {@JsonSchemaInt(path = "multipleOf", value = 1000)})
    @Max(200000)
    @JsonProperty
    private int miles;

    @Min(2000)
    @JsonProperty
    private int year;

    @JsonProperty
    private Date purchaseDate;

    @JsonProperty
    @JsonFormat(shape = JsonFormat.Shape.NUMBER)
    private Date listedDate;

    @JsonProperty
    private String[] owners;

    @JsonProperty
    private Collection<Float> serviceChecks;

    // Empty constructor is required by Jackson to deserialize bytes
    // into an Object of this class
    public Car() {}
}
```

Erstellen eines Schemas

Sie können ein Schema mit dem erstellen AWS Glue APIs oder das AWS Glue console.

AWS Glue APIs

Sie können diese Schritte verwenden, um diese Aufgabe mithilfe der AWS Glue APIs.

Um ein neues Schema hinzuzufügen, verwenden Sie die CreateSchema Aktion (Python: create_schema)-API.

Geben Sie eine `RegistryId`-Struktur an, um eine Registrierung für das Schema anzugeben. Oder lassen Sie die `RegistryId` weg, um die Standardregistrierung zu verwenden.

Geben Sie einen `SchemaName` bestehend aus Buchstaben, Ziffern, Bindestrichen oder Unterstrichen an, und geben Sie als `DataFormat` **`AVRO`** oder **`JSON`** an. Ist das `DataFormat` einmal für ein Schema festgelegt, ist nicht änderbar.

Geben Sie einen `Compatibility`-Modus an:

- Backward (Abwärts) (empfohlen) – Der Verbraucher kann sowohl die aktuelle als auch die vorherige Version lesen.
- Backward all (Abwärts alle) – Der Verbraucher kann die aktuelle und alle früheren Versionen lesen.
- Forward (Aufwärts) – Der Verbraucher kann sowohl die aktuelle als auch die nachfolgende Version lesen.
- Forward all (Aufwärts alle) – Der Verbraucher kann sowohl die aktuelle als auch alle nachfolgenden Versionen lesen.
- Full (Voll) – Kombination aus „Backward“ (Abwärts) und „Forward“ (Aufwärts).
- Full all (Voll alle) – Kombination aus „Backward all“ (Abwärts alle) und „Forward all“ (Aufwärts alle).
- None (Keine) – Es werden keine Kompatibilitätsprüfungen durchgeführt.
- Disabled (Deaktiviert) – Verhindert das Versioning für dieses Schema.

Geben Sie optional `Tags` für Ihr Schema an.

Geben Sie eine `SchemaDefinition` an, um das Schema im Avro-, JSON-oder Protobuf-Datenformat zu definieren. Informationen finden Sie in den Beispielen.

Für Avro-Datenformat:

```
aws glue create-schema --registry-id RegistryName="registryName1" --schema-name
 testschema --compatibility NONE --data-format AVRO --schema-definition "{\"type\":
 \"record\", \"name\": \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"},
 {\"name\": \"f2\", \"type\": \"string\"} ]}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1" --schema-name testschema --compatibility
 NONE --data-format AVRO  --schema-definition "{\"type\": \"record\", \"name\": \"r1\",
 \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type\":
 \"string\"} ]}"
```

Für JSON-Datenformat:

```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaJson --compatibility NONE --data-format JSON --schema-definition "{\"$schema
\": \"http://json-schema.org/draft-07/schema#\",\"type\":\"object\",\"properties\":
{\"f1\":{\"type\":\"string\"}}}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaJson --compatibility
 NONE --data-format JSON --schema-definition "{\"$schema\": \"http://json-schema.org/
draft-07/schema#\",\"type\":\"object\",\"properties\":{\"f1\":{\"type\":\"string\"}}}"
```

Für das Protobuf-Datenformat:

```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaProtobuf --compatibility NONE --data-format PROTOBUF --schema-definition
 "syntax = \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaProtobuf
 --compatibility NONE --data-format PROTOBUF --schema-definition "syntax =
 \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

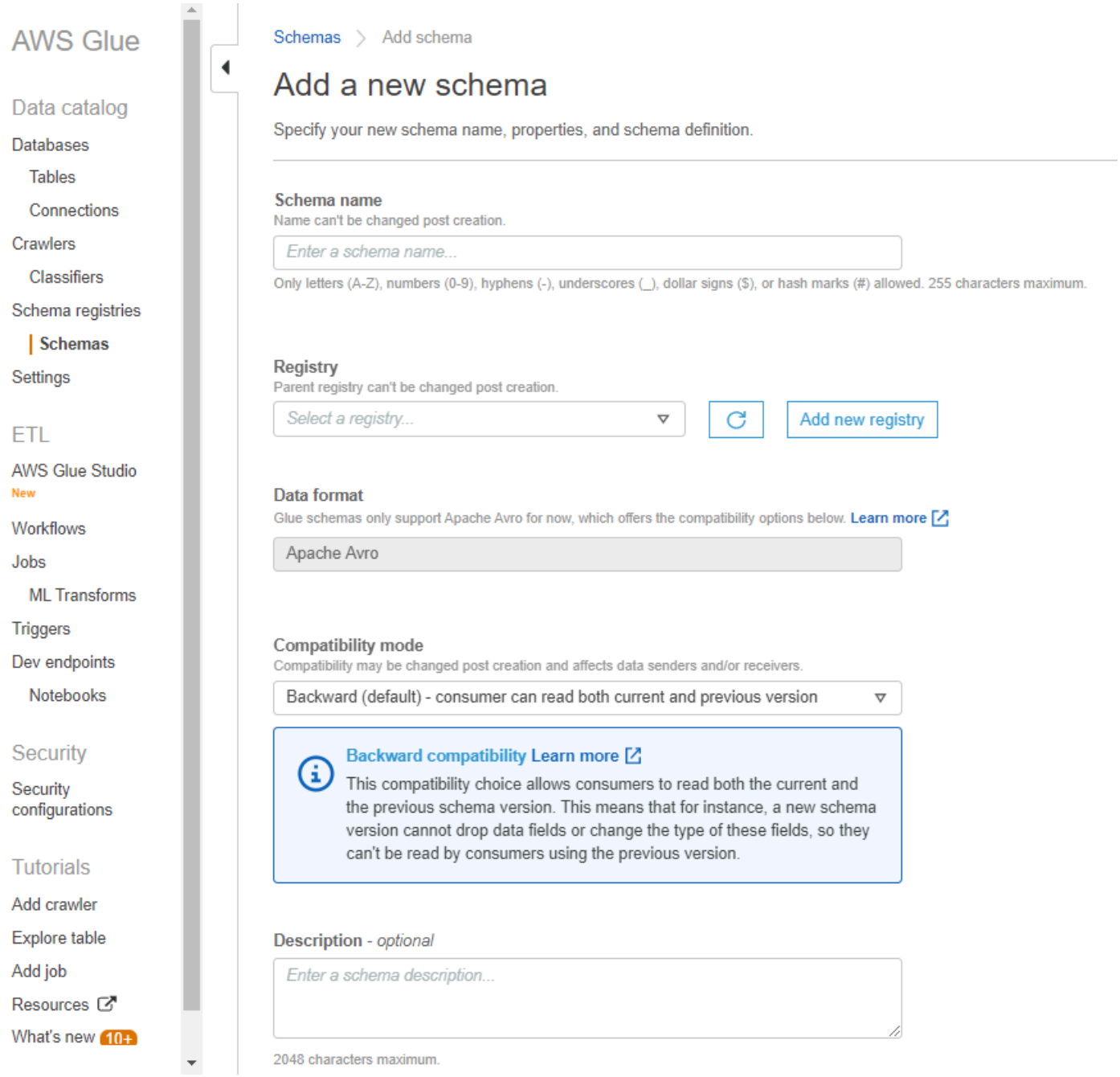
AWS Glue Konsole

Um ein neues Schema hinzuzufügen, verwenden Sie den AWS Glue Konsole:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schemas (Schemata).
3. Klicken Sie auf Add Schema (Schema hinzufügen).
4. Geben Sie einen Schemanamen an, bestehend aus Buchstaben, Zahlen, Bindestrichen, Unterstrichen, Dollarzeichen oder Hashmarks. Dieser Name kann nicht geändert werden.
5. Wählen Sie im Dropdown-Menü die Registry, in der das Schema gespeichert wird. Die übergeordnete Registrierung kann nach der Erstellung nicht geändert werden.
6. Lassen Sie das Data format (Datenformat) als Apache Avro oder JSON. Dieses Format gilt für alle Versionen dieses Schemas.
7. Wählen Sie einen Compatibility mode (Kompatibilitätsmodus).

- Backward (Abwärts) (empfohlen) – Receiver kann sowohl aktuelle als auch frühere Versionen lesen.
- Backward All (Abwärts alle) – Receiver kann aktuelle und alle früheren Versionen lesen.
- Vorwärts – Absender kann sowohl aktuelle als auch frühere Versionen schreiben.
- Vorwärts alle – Absender kann sowohl aktuelle als auch alle früheren Versionen schreiben.
- Full (Voll) – Kombination aus „Backward“ (Abwärts) und „Forward“ (Aufwärts).
- Full all (Voll alle) – Kombination aus „Backward all“ (Abwärts alle) und „Forward all“ (Aufwärts alle).
- Keine – Es werden keine Kompatibilitätsprüfungen durchgeführt.
- Disabled (Deaktiviert) – Verhindert das Versioning für dieses Schema.

8. Geben Sie eine optionale Beschreibung für die Registrierung ein (max. 250 Zeichen).

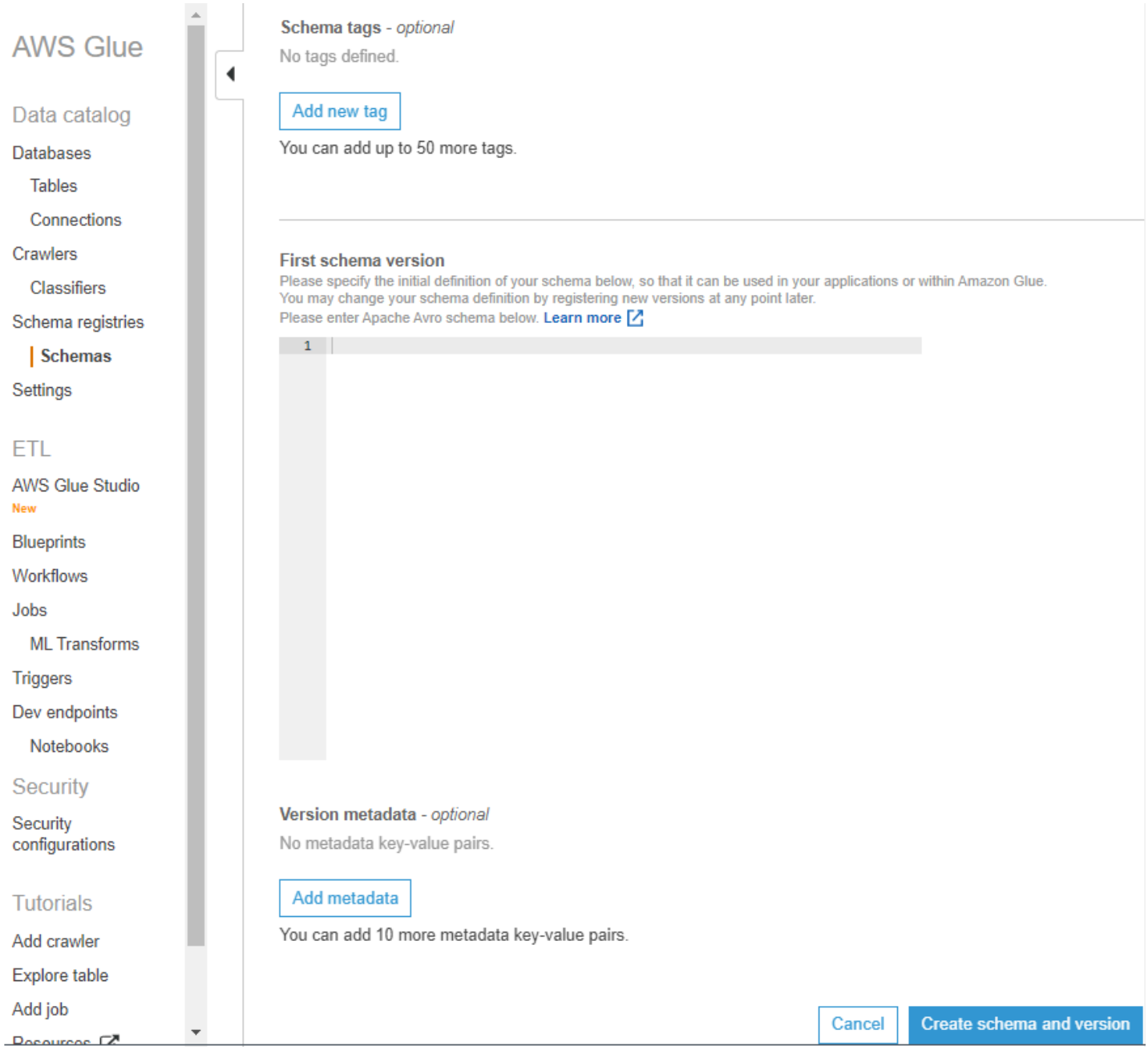


9. Wenden Sie optional ein oder mehrere Tags auf Ihr Schema an. Klicken Sie auf Add new Tag (Neuen Tag hinzufügen) und geben Sie einen Tag-Schlüssel und optional einen Tag-Wert ein.
10. Geben Sie in das Feld Erste Schemaversion das ursprüngliche Schema ein oder fügen Sie es ein.

    Informationen zum Avro-Format finden Sie unter Arbeiten mit dem Avro-Datenformat.

    Informationen zum JSON-Format finden Sie unter Arbeiten mit dem JSON-Datenformat.

11. Wählen Sie alternativ die Option Add metadata (Metadaten hinzufügen), um Versionsmetadaten hinzuzufügen und Ihre Schemaversion mit Anmerkungen zu versehen oder zu klassifizieren.
12. Klicken Sie auf Create schema and version (Schema und Version erstellen).



Das Schema wird erstellt und in der Liste unter Schemas (Schemata) angezeigt.

Arbeiten mit dem Avro-Datenformat

Avro bietet Datenserialisierung und Datenaustausch. Avro speichert die Datendefinition im JSON-Format, wodurch es leicht zu lesen und zu interpretieren ist. Die Daten selbst werden im Binärformat gespeichert.

Informationen zum Definieren eines Apache-Avro-Schemas finden Sie unter Apache Avro-Spezifikationen.

Arbeiten mit dem JSON-Datenformat

Daten können im JSON-Format serialisiert werden. Das JSON-Schemaformat definiert den Standard für das Format des JSON-Schemas.

Aktualisieren eines Schemas oder einer Registrierung

Nach der Erstellung können Sie Schemata, Schemaversionen und die Registrierung bearbeiten.

Aktualisieren einer Registrierung

Sie können eine Registrierung aktualisieren, indem Sie AWS Glue APIs oder das AWS Glue console. Der Name einer vorhandenen Registrierung kann nicht bearbeitet werden. Sie können die Beschreibung für eine Registrierung bearbeiten.

AWS Glue APIs

Um eine vorhandene Registrierung zu aktualisieren, verwenden Sie die UpdateRegistry Aktion (Python: update_registry)-API.

Wählen Sie eine `RegistryId`-Struktur, um die Registrierung anzugeben, die Sie aktualisieren möchten. Übergeben Sie eine `Description`, um die Beschreibung für eine Registrierung zu ändern.

```
aws glue update-registry --description updatedDescription --registry-id
 RegistryArn="arn:aws:glue:us-east-2:901234567890:registry/registryName1"
```

AWS Glue Konsole

Um eine Registrierung mit dem zu aktualisieren AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.

2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schema Registries (Schemaregistrierungen).
3. Wählen Sie eine Registrierung aus der Liste der Registrierungen, indem Sie das entsprechende Kontrollkästchen aktivieren.
4. Klicken Sie im Menü Action (Aktion) auf Edit registry (Registrierung bearbeiten).

Aktualisieren eines Schemas

Sie können die Beschreibung und die Kompatibilitätseinstellung für ein Schema aktualisieren.

Um ein vorhandenes Schema zu aktualisieren, verwenden Sie die UpdateSchema Aktion (Python: update_schema)-API.

Wählen Sie eine `SchemaId`-Struktur, um das Schema anzugeben, die Sie aktualisieren möchten. Eine `VersionNumber` oder `Compatibility` muss angegeben werden.

Codebeispiel 11:

```
aws glue update-schema --description testDescription --schema-id
 SchemaName="testSchema1",RegistryName="registryName1" --schema-version-number
 LatestVersion=true --compatibility NONE
```

```
aws glue update-schema --description testDescription --schema-id
 SchemaArn="arn:aws:glue:us-east-2:901234567890:schema/registryName1/testSchema1" --
schema-version-number LatestVersion=true --compatibility NONE
```

Eine Schemaversion hinzufügen

Wenn Sie eine Schemaversion hinzufügen, müssen Sie die Versionen vergleichen, um sicherzustellen, dass das neue Schema akzeptiert wird.

Um eine neue Version zu einem vorhandenen Schema hinzuzufügen, verwenden Sie die RegisterSchemaVersion Aktion (Python: register_schema_version)-API.

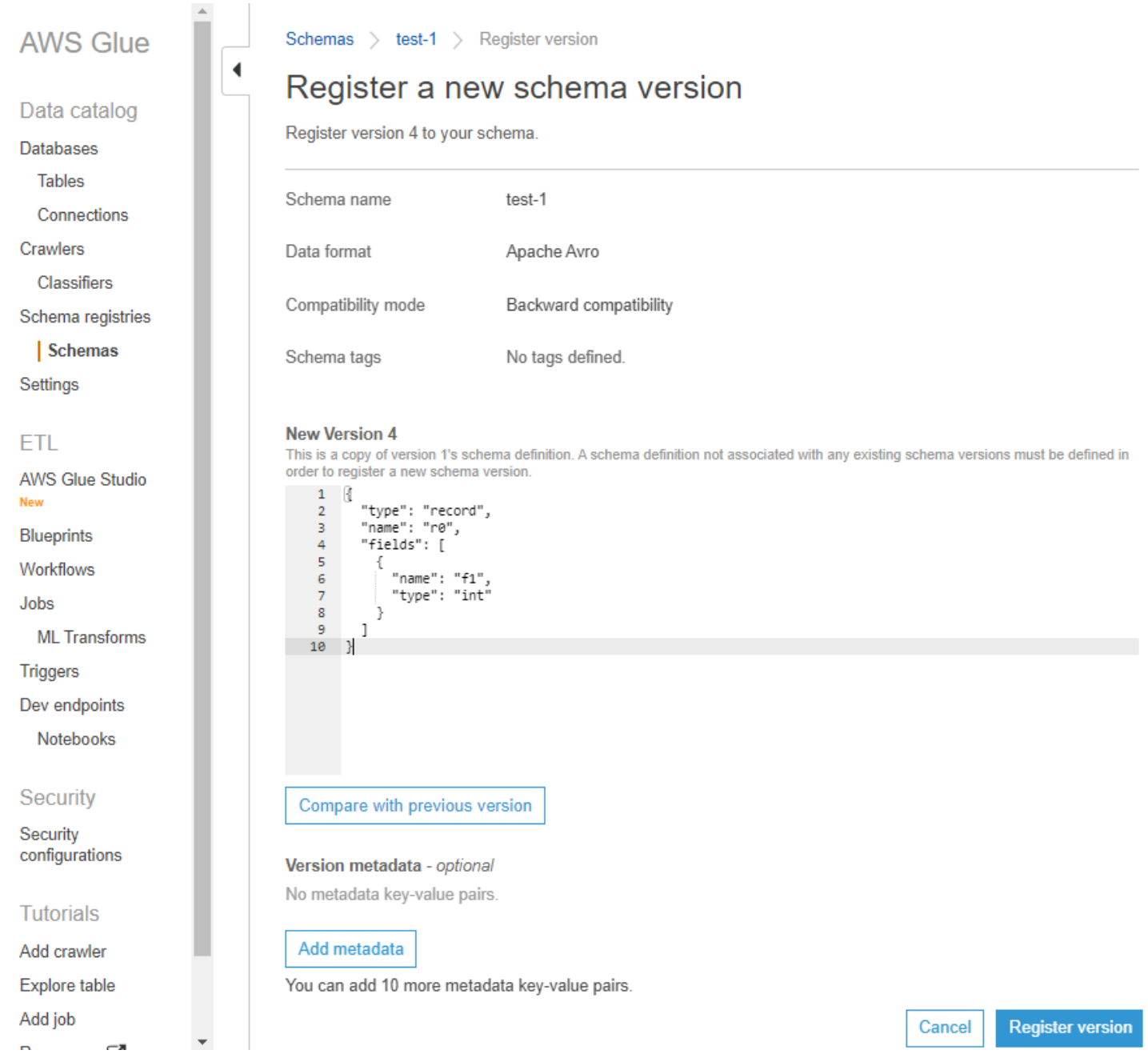
Wählen Sie eine `SchemaId`-Struktur, um das Schema anzugeben, für das Sie eine Version hinzufügen möchten, und eine `SchemaDefinition`, um das Schema zu definieren.

Codebeispiel 12:

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
\": \"string\"} ]}" --schema-id SchemaArn="arn:aws:glue:us-east-1:901234567890:schema/
registryName/testschema"
```

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
\": \"string\"} ]}" --schema-id SchemaName="testschema",RegistryName="testregistry"
```

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schemas (Schemata).
3. Wählen Sie das Schema aus der Liste der Schemata aus, indem Sie das entsprechende Kontrollkästchen aktivieren.
4. Wählen Sie ein oder mehrere Schemata in der Liste aus, indem Sie die Kontrollkästchen aktivieren.
5. Wählen Sie im Menü Action (Aktion) die Option Register a new version (Neue Version registrieren).
6. Geben Sie das neue Schema in das Feld New version (Neue Version) ein oder fügen Sie es ein.
7. Klicken Sie auf Compare with previous version (Vergleich mit vorheriger Version), um Unterschiede zur vorherigen Schemaversion anzuzeigen.
8. Wählen Sie alternativ die Option Add metadata (Metadaten hinzufügen), um Versionsmetadaten hinzuzufügen und Ihre Schemaversion mit Anmerkungen zu versehen oder zu klassifizieren. Geben Sie den Schlüssel und optional einen Wert ein.
9. Klicken Sie auf Register version (Version registrieren).



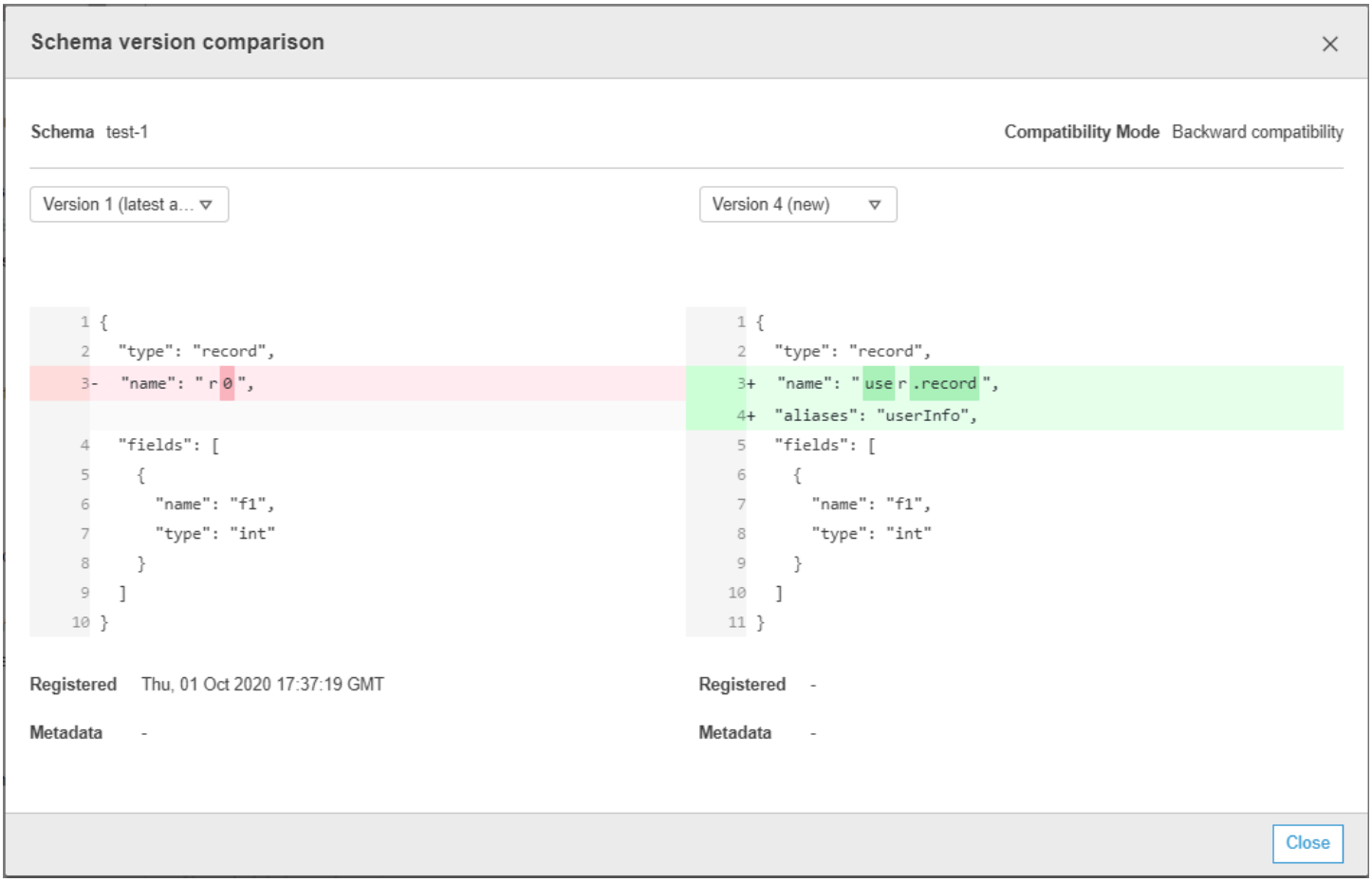
Die Version des Schemas (der Schemata) wird in der Liste der Versionen angezeigt. Wenn die Version den Kompatibilitätsmodus geändert hat, wird die Version als Checkpoint markiert.

Beispiel für einen Vergleich von Schemaversionen

Wenn Sie entscheiden, die Option Compare with previous version (Vergleich mit vorheriger Version) zu verwenden, werden die vorherigen und die neuen Versionen zusammen angezeigt. Geänderte Informationen werden wie folgt hervorgehoben:

- Gelb: Kennzeichnet geänderte Informationen.
- Grün: Kennzeichnet Inhalte, die in der neuesten Version hinzugefügt wurden.
- Rot: Kennzeichnet Inhalte, die in der neuesten Version entfernt wurden.

Sie können auch einen Vergleich mit früheren Versionen durchführen.



Ein Schema oder eine Registrierung löschen

Das Löschen eines Schemas, einer Schemaversion oder einer Registrierung ist eine permanente Aktion, die nicht rückgängig gemacht werden kann.

Ein Schema löschen

Möglicherweise möchten Sie ein Schema löschen, wenn es nicht mehr in einer Registrierung verwendet werden soll, indem Sie die AWS Management Console oder die DeleteSchema Aktion (Python: delete_schema) API verwenden.

Das Löschen eines oder mehrerer Schemata ist ein dauerhafter Vorgang, der nicht rückgängig gemacht werden kann. Stellen Sie sicher, dass das Schema oder die Schemata nicht mehr benötigt werden.

Um ein Schema aus der Registrierung zu löschen, rufen Sie die DeleteSchema Aktion (Python: delete_schema)-API auf und geben die `SchemaId`-Struktur an, um das Schema zu identifizieren.

Zum Beispiel:

```
aws glue delete-schema --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/registryName1/schemaname"
```

```
aws glue delete-schema --schema-id SchemaName="TestSchema6-
deleteschemabyname",RegistryName="default-registry"
```

AWS Glue Konsole

Um ein Schema aus dem zu löschen AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schema Registries (Schemaregistrierungen).
3. Wählen Sie die Registrierung, die Ihr Schema enthält, aus der Liste der Registrierungen aus.
4. Wählen Sie ein oder mehrere Schemata in der Liste aus, indem Sie die Kontrollkästchen aktivieren.
5. Wählen Sie im Menü Actions (Aktionen) die Option Delete (Löschen).
6. Geben Sie **`Delete`** in das Feld ein, um den Löschvorgang zu bestätigen.
7. Wählen Sie Löschen.

Die von Ihnen angegebenen Schemata werden aus der Registrierung gelöscht.

Eine Schemaversion löschen

Da sich Schemas in der Registrierung ansammeln, möchten Sie möglicherweise unerwünschte Schemaversionen mithilfe der AWS Management Console oder der DeleteSchemaVersions Aktion

(Python: delete_schema_versions) API löschen. Das Löschen einer oder mehrerer Schemaversionen ist ein dauerhafter Vorgang, der nicht rückgängig gemacht werden kann. Stellen Sie sicher, dass die Schemaversionen nicht mehr benötigt werden.

Beachten Sie beim Löschen von Schemaversionen die folgenden Einschränkungen:

- Sie können keine Version mit Checkpoint löschen.
- Die Anzahl zusammenhängender Versionen darf nicht mehr als 25 betragen.
- Die neueste Schemaversion darf sich nicht im Status „Ausstehend“ befinden.

Geben Sie die `SchemaId`-Struktur an, um das Schema zu identifizieren, und geben Sie `Versions` als Bereich von Versionen an, die gelöscht werden sollen. Weitere Informationen zum Angeben einer Version oder eines Versionsbereichs finden Sie unter DeleteRegistry Aktion (Python: delete_registry). Die von Ihnen angegebenen Schemaversionen werden aus der Registrierung gelöscht.

Das Aufrufen der ListSchemaVersions Aktion (Python: list_schema_versions)-API nach diesem Aufruf listet den Status der gelöschten Versionen auf.

Zum Beispiel:

```
aws glue delete-schema-versions --schema-id
 SchemaName="TestSchema6",RegistryName="default-registry" --versions "1-1"
```

```
aws glue delete-schema-versions --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/default-registry/TestSchema6-NON-Existent" --versions "1-1"
```

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schema Registries (Schemaregistrierungen).
3. Wählen Sie die Registrierung, die Ihr Schema enthält, aus der Liste der Registrierungen aus.
4. Wählen Sie ein oder mehrere Schemata in der Liste aus, indem Sie die Kontrollkästchen aktivieren.
5. Wählen Sie im Menü Actions (Aktionen) die Option Delete (Löschen).
6. Geben Sie **`Delete`** in das Feld ein, um den Löschvorgang zu bestätigen.
7. Wählen Sie Löschen.

Die von Ihnen angegebenen Schemaversionen werden aus der Registrierung gelöscht.

Eine Registrierung löschen

Unter Umständen möchten Sie eine Registrierung löschen, wenn die darin enthaltenen Schemata nicht mehr unter dieser Registrierung organisiert werden sollen. Sie müssen diese Schemata einer anderen Registrierung zuweisen.

Das Löschen einer oder mehrerer Registrierungen ist ein dauerhafter Vorgang, der nicht rückgängig gemacht werden kann. Stellen Sie sicher, dass die Registrierung(en) nicht mehr benötigt wird/werden.

Die Standardregistrierung kann mit der AWS CLI gelöscht werden.

AWS Glue API

Um die gesamte Registrierung einschließlich Schema und aller Schemaversionen zu löschen, rufen Sie die DeleteRegistry Aktion (Python: delete_registry)-API auf. Geben Sie eine `RegistryId`-Struktur an, um die Registrierung zu identifizieren.

Zum Beispiel:

```
aws glue delete-registry --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1"
```

```
aws glue delete-registry --registry-id RegistryName="TestRegistry-deletebyname"
```

Um den Status des Löschvorgangs zu erhalten, können Sie die `GetRegistry`-API nach dem asynchronen Aufruf aufrufen.

AWS Glue Konsole

Um eine Registrierung aus dem zu löschen AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Klicken Sie im Navigationsbereich unter Data Catalog auf Schema Registries (Schemaregistrierungen).
3. Wählen Sie eine Registrierung aus der Liste, indem Sie das entsprechende Kontrollkästchen aktivieren.
4. Klicken Sie im Menü Action (Aktion) auf Delete registry (Registrierung löschen).

5. Geben Sie **`Delete`** in das Feld ein, um den Löschvorgang zu bestätigen.
6. Wählen Sie Löschen.

Die von Ihnen ausgewählten Registrierungen wurden gelöscht von AWS Glue.

IAM-Beispiele für Serialisierer

> **Note**
>
> AWS verwaltete Richtlinien gewähren die erforderlichen Berechtigungen für allgemeine Anwendungsfälle. Informationen zur Verwendung verwalteter Richtlinien zum Verwalten der Schemaregistrierung finden Sie unter AWS verwaltete (vordefinierte) Richtlinien für AWS Glue.

Für Serialisierer sollten Sie eine minimale Richtlinie wie unten erstellen, um eine Möglichkeit zu haben, die `schemaVersionId` für eine bestimmte Schemadefinition zu finden. Achten Sie darauf, dass Sie Leserechte für die Registrierung haben, um die Schemata in der Registrierung zu lesen. Sie können die Registrierungen, die gelesen werden können, mit der `Resource`-Klausel einschränken.

Codebeispiel 13:

```
{
    "Sid" : "GetSchemaByDefinition",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition"
    ],
        "Resource" : ["arn:aws:glue:us-east-2:012345678:registry/registryname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-2"
                     ]
}
```

Sie können Produzenten auch erlauben, neue Schemata und Versionen zu erstellen, indem Sie die folgenden zusätzlichen Methoden einbeziehen. Beachten Sie, dass Sie in der Lage sein sollten, die

Registrierung anhand add/remove/evolve der darin enthaltenen Schemas zu überprüfen. Sie können die Registrierungen, die inspiziert werden können, mit der `Resource`-Klausel einschränken.

Codebeispiel 14:

```
{
    "Sid" : "RegisterSchemaWithMetadata",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition",
        "glue:CreateSchema",
        "glue:RegisterSchemaVersion",
        "glue:PutSchemaVersionMetadata",
    ],
    "Resource" : ["arn:aws:glue:aws-region:123456789012:registry/registryname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-2"
                ]
}
```

IAM-Beispiele für Deserialisierer

Für Deserialisierer (Verbraucherseite) sollten Sie eine ähnliche Richtlinie wie unten erstellen, damit der Deserializer das Schema aus der Schemaregistrierung zur Deserialisierung abrufen kann. Beachten Sie, dass Sie in der Lage sein sollten, die Registrierung zu überprüfen, um Schemata darin abzurufen.

Codebeispiel 15:

```
{
    "Sid" : "GetSchemaVersion",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaVersion"
    ],
    "Resource" : ["*"]
}
```

Private Konnektivität mit AWS PrivateLink

Sie können AWS PrivateLink es verwenden, um die VPC Ihres Datenproduzenten zu verbinden AWS Glue durch Definition eines Schnittstellen-VPC-Endpunkts für AWS Glue. Wenn Sie einen VPC-Schnittstellenendpunkt verwenden, erfolgt die Kommunikation zwischen Ihrer VPC und AWS Glue wird vollständig innerhalb des Netzwerks abgewickelt. AWS Weitere Informationen finden Sie unter Verwenden AWS Glue mit VPC-Endpunkten.

Zugreifen auf CloudWatch Amazon-Metriken

CloudWatch Amazon-Metriken sind im Rahmen CloudWatch des kostenlosen Kontingents verfügbar. Sie können in der CloudWatch Konsole auf diese Messwerte zugreifen. Zu den Metriken auf API-Ebene gehören CreateSchema (Erfolg und Latenz) GetSchemaByDefinition, (Erfolg und Latenz), GetSchemaVersion (Erfolg und Latenz), RegisterSchemaVersion (Erfolg und Latenz), PutSchemaVersionMetadata (Erfolg und Latenz). Zu den Metriken auf Ressourcenebene gehört die Registrierung. ThrottledByLimit,. SchemaVersion ThrottledByLimit, SchemaVersion. Größe.

AWS CloudFormation Beispielvorlage für die Schemaregistrierung

Im Folgenden finden Sie eine Beispielvorlage für die Erstellung von Schemaregistrierungsressourcen in AWS CloudFormation. Um diesen Stack in Ihrem Konto zu erstellen, kopieren Sie die obige Vorlage in eine Datei `SampleTemplate.yaml` und führen Sie dann den folgenden Befehl aus:

```
aws cloudformation create-stack --stack-name ABCSchemaRegistryStack --template-body
 "'cat SampleTemplate.yaml'"
```

In diesem Beispiel verwenden wir die `AWS::Glue::Registry`, um eine Registrierung zu erstellen, `AWS::Glue::Schema` um ein Schema zu erstellen, `AWS::Glue::SchemaVersion` um eine Schemaversion zu erstellen und `AWS::Glue::SchemaVersionMetadata` um die Schemaversion mit Metadaten zu füllen.

```
Description: "A sample CloudFormation template for creating Schema Registry resources."
Resources:
  ABCRegistry:
    Type: "AWS::Glue::Registry"
    Properties:
      Name: "ABCSchemaRegistry"
      Description: "ABC Corp. Schema Registry"
      Tags:
```

```
        - Key: "Project"
          Value: "Foo"
  ABCSchema:
    Type: "AWS::Glue::Schema"
    Properties:
      Registry:
        Arn: !Ref ABCRegistry
      Name: "TestSchema"
      Compatibility: "NONE"
      DataFormat: "AVRO"
      SchemaDefinition: >
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"name","type":"string"},{"name":"favorite_number","type":"int"}]}
      Tags:
        - Key: "Project"
          Value: "Foo"
  SecondSchemaVersion:
    Type: "AWS::Glue::SchemaVersion"
    Properties:
      Schema:
        SchemaArn: !Ref ABCSchema
      SchemaDefinition: >
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"status","type":"string", "default":"ON"}, {"name":"name","type":"string"},
{"name":"favorite_number","type":"int"}]}
  FirstSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !GetAtt ABCSchema.InitialSchemaVersionId
      Key: "Application"
      Value: "Kinesis"
  SecondSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !Ref SecondSchemaVersion
      Key: "Application"
      Value: "Kinesis"
```

Integration in AWS Glue Schema Registry

In diesen Abschnitten werden Integrationen mit beschrieben AWS Glue Schemaregistrierung. Die Beispiele in diesem Abschnitt zeigen ein Schema mit AVRO-Datenformat. Weitere Beispiele,

einschließlich Schemas mit dem JSON-Datenformat, finden Sie in den Integrationstests und ReadMe Informationen im AWS Glue Open-Source-Repository für die Schemaregistrierung.

Themen

- Anwendungsfall: Verbinden der Schema Registry mit Amazon MSK oder Apache Kafka
- Anwendungsfall: Integration von Amazon Kinesis Data Streams mit AWS Glue Schema Registry
- Anwendungsfall: Amazon Managed Service für Apache Flink
- Anwendungsfall: Integration mit AWS Lambda
- Anwendungsfall: AWS Glue Data Catalog
- Anwendungsfall: AWS Glue Streamen
- Anwendungsfall: Apache Kafka Streams

Anwendungsfall: Verbinden der Schema Registry mit Amazon MSK oder Apache Kafka

Nehmen wir an, Sie schreiben Daten in ein Apache-Kafka-Thema, und Sie können diese Schritte ausführen, um loszulegen.

1. Erstellen Sie einen Amazon Managed Streaming for Apache Kafka (Amazon MSK) oder Apache Kafka-Cluster mit mindestens einem Thema. Wenn Sie einen Amazon-MSK-Cluster erstellen, können Sie die AWS Management Console verwenden. Folgen Sie den Anweisungen unter Erste Schritte mit Amazon MSK im Entwicklerhandbuch für Amazon Managed Streaming for Apache Kafka.
2. Folgen Sie dem Schritt SerDe Bibliotheken installieren oben.
3. Um Schemaregistrierungen, Schemata oder Schemaversionen zu erstellen, befolgen Sie die Anweisungen im Abschnitt Erste Schritte mit der Schemaregistrierung dieses Dokuments.
4. Motivieren Sie Ihre Produzenten und Verbraucher, die Schema Registry zu verwenden, um Datensätze in Amazon-MSK- oder Apache-Kafka-Themen zu schreiben und daraus zu lesen. Ein Beispiel für Produzenten- und Consumer-Code finden Sie in der ReadMe Datei aus den Serde-Bibliotheken. Die Schema-Registry-Bibliothek des Produzenten serialisiert den Datensatz automatisch und versieht den Datensatz mit einer Schemaversions-ID.
5. Wenn das Schema dieses Datensatzes eingegeben wurde oder die automatische Registrierung aktiviert ist, ist das Schema in der Schema Registry registriert.
6. Der Verbraucher liest aus dem Amazon MSK- oder Apache Kafka-Thema und verwendet dabei den AWS Glue Die Schema Registry-Bibliothek sucht das Schema automatisch in der Schema Registry.

Anwendungsfall: Integration von Amazon Kinesis Data Streams mit AWS Glue Schema Registry

Diese Integration erfordert, dass Sie einen vorhandenen Amazon Kinesis Data Stream haben. Weitere Informationen finden Sie unter Erste Schritte mit Amazon Kinesis Data Streams im Entwicklerhandbuch für Amazon Kinesis Data Streams.

Es gibt zwei Möglichkeiten, mit Daten in einem Kinesis-Datenstrom zu interagieren.

- Über die Bibliotheken Kinesis Producer Library (KPL) und Kinesis Client Library (KCL) in Java. Mehrsprachige Unterstützung wird nicht bereitgestellt.
- Durch die`PutRecords`,`PutRecord`, und `GetRecords` Kinesis Data Streams, die in der APIs AWS SDK für Java verfügbar sind.

Wenn Sie derzeit die KPL/KCL-Bibliotheken verwenden, empfehlen wir, diese Methode weiterhin zu verwenden. Es gibt aktualisierte KCL- und KPL-Versionen mit integrierter Schema Registry, wie in den Beispielen gezeigt. Andernfalls können Sie den Beispielcode verwenden, um Folgendes zu nutzen AWS Glue Schemaregistrierung, wenn Sie das KDS APIs direkt verwenden.

Die Integration der Schema Registry ist mit KPL v0.14.2 oder höher und mit KCL v2.3 oder höher verfügbar. Die Integration der Schema Registry mit dem JSON-Datenformat ist mit KPL v0.14.8 oder höher und mit KCL v2.3.6 oder höher verfügbar.

Interaktion mit Daten mit Kinesis SDK V2

In diesem Abschnitt wird die Interaktion mit Kinesis mithilfe des Kinesis SDK V2 beschrieben

```
// Example JSON Record, you can construct a AVRO record also
private static final JsonDataWithSchema record =
 JsonDataWithSchema.builder(schemaString, payloadString);
private static final DataFormat dataFormat = DataFormat.JSON;

//Configurations for Schema Registry
GlueSchemaRegistryConfiguration gsrConfig = new GlueSchemaRegistryConfiguration("us-
east-1");

GlueSchemaRegistrySerializer glueSchemaRegistrySerializer =
        new GlueSchemaRegistrySerializerImpl(awsCredentialsProvider, gsrConfig);
GlueSchemaRegistryDataFormatSerializer dataFormatSerializer =
        new GlueSchemaRegistrySerializerFactory().getInstance(dataFormat, gsrConfig);
```

```
Schema gsrSchema =
        new Schema(dataFormatSerializer.getSchemaDefinition(record), dataFormat.name(),
 "MySchema");

byte[] serializedBytes = dataFormatSerializer.serialize(record);

byte[] gsrEncodedBytes = glueSchemaRegistrySerializer.encode(streamName, gsrSchema,
 serializedBytes);

PutRecordRequest putRecordRequest = PutRecordRequest.builder()
        .streamName(streamName)
        .partitionKey("partitionKey")
        .data(SdkBytes.fromByteArray(gsrEncodedBytes))
        .build();
shardId = kinesisClient.putRecord(putRecordRequest)
        .get()
        .shardId();

GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer = new
 GlueSchemaRegistryDeserializerImpl(awsCredentialsProvider, gsrConfig);

GlueSchemaRegistryDataFormatDeserializer gsrDataFormatDeserializer =
        glueSchemaRegistryDeserializerFactory.getInstance(dataFormat, gsrConfig);

GetShardIteratorRequest getShardIteratorRequest = GetShardIteratorRequest.builder()
        .streamName(streamName)
        .shardId(shardId)
        .shardIteratorType(ShardIteratorType.TRIM_HORIZON)
        .build();

String shardIterator = kinesisClient.getShardIterator(getShardIteratorRequest)
        .get()
        .shardIterator();

GetRecordsRequest getRecordRequest = GetRecordsRequest.builder()
        .shardIterator(shardIterator)
        .build();
GetRecordsResponse recordsResponse = kinesisClient.getRecords(getRecordRequest)
        .get();

List<Object> consumerRecords = new ArrayList<>();
List<Record> recordsFromKinesis = recordsResponse.records();

for (int i = 0; i < recordsFromKinesis.size(); i++) {
```

```
    byte[] consumedBytes = recordsFromKinesis.get(i)
            .data()
            .asByteArray();

    Schema gsrSchema = glueSchemaRegistryDeserializer.getSchema(consumedBytes);
    Object decodedRecord =
 gsrDataFormatDeserializer.deserialize(ByteBuffer.wrap(consumedBytes),

 gsrSchema.getSchemaDefinition());
    consumerRecords.add(decodedRecord);
}
```

Interaktion mit Daten mithilfe der KPL/KCL-Bibliotheken

In diesem Abschnitt wird die Integration von Kinesis Data Streams mit Schema Registry mithilfe von beschriebenKPL/KCL libraries. For more information on using KPL/KCL, siehe Developing Producer Using the Amazon Kinesis Producer Library im Amazon Kinesis Data Streams Developer Guide.

Einrichten der Schema Registry in KPL

1. Definieren Sie die Schemadefinition für die Daten, das Datenformat und den Schemanamen, wie sie in der AWS Glue Schemaregistrierung.
2. Konfigurieren Sie optional das `GlueSchemaRegistryConfiguration`-Objekt.
3. Übergeben Sie das Schemaobjekt an die `addUserRecord API`.

```
private static final String SCHEMA_DEFINITION = "{"namespace": "example.avro",\n"
+ " "type": "record",\n"
+ " "name": "User",\n"
+ " "fields": [\n"
+ " {"name": "name", "type": "string"},\n"
+ " {"name": "favorite_number", "type": ["int", "null"]},\n"
+ " {"name": "favorite_color", "type": ["string", "null"]}\n"
+ " ]\n"
+ "}";

KinesisProducerConfiguration config = new KinesisProducerConfiguration();
config.setRegion("us-west-1")

//[Optional] configuration for Schema Registry.

GlueSchemaRegistryConfiguration schemaRegistryConfig =
new GlueSchemaRegistryConfiguration("us-west-1");
```

```
schemaRegistryConfig.setCompression(true);

config.setGlueSchemaRegistryConfiguration(schemaRegistryConfig);

///Optional configuration ends.

final KinesisProducer producer =
      new KinesisProducer(config);

final ByteBuffer data = getDataToSend();

com.amazonaws.services.schemaregistry.common.Schema gsrSchema =
    new Schema(SCHEMA_DEFINITION, DataFormat.AVRO.toString(), "demoSchema");

ListenableFuture<UserRecordResult> f = producer.addUserRecord(
config.getStreamName(), TIMESTAMP, Utils.randomExplicitHashKey(), data, gsrSchema);

private static ByteBuffer getDataToSend() {
      org.apache.avro.Schema avroSchema =
        new org.apache.avro.Schema.Parser().parse(SCHEMA_DEFINITION);

      GenericRecord user = new GenericData.Record(avroSchema);
      user.put("name", "Emily");
      user.put("favorite_number", 32);
      user.put("favorite_color", "green");

      ByteArrayOutputStream outBytes = new ByteArrayOutputStream();
      Encoder encoder = EncoderFactory.get().directBinaryEncoder(outBytes, null);
      new GenericDatumWriter<>(avroSchema).write(user, encoder);
      encoder.flush();
      return ByteBuffer.wrap(outBytes.toByteArray());
 }
```

Einrichten der Kinesis Client-Bibliothek

Sie entwickeln Ihre Kinesis Client Library-Verbraucher in Java. Weitere Informationen finden Sie unter Entwickeln eines Kinesis Client Library-Verbrauchers in Java im Entwicklerhandbuch zu Amazon Kinesis Data Streams.

1. Erstellen Sie eine Instance von `GlueSchemaRegistryDeserializer` durch Übergeben eines `GlueSchemaRegistryConfiguration`-Objekts.

2. Übergeben Sie den `GlueSchemaRegistryDeserializer` an `retrievalConfig.glueSchemaRegistryDeserializer`.
3. Greifen Sie auf das Schema eingehender Nachrichten zu, indem Sie `kinesisClientRecord.getSchema()` aufrufen.

```
GlueSchemaRegistryConfiguration schemaRegistryConfig =
    new GlueSchemaRegistryConfiguration(this.region.toString());

 GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer =
    new
 GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
 schemaRegistryConfig);

 RetrievalConfig retrievalConfig =
 configsBuilder.retrievalConfig().retrievalSpecificConfig(new
 PollingConfig(streamName, kinesisClient));
 retrievalConfig.glueSchemaRegistryDeserializer(glueSchemaRegistryDeserializer);

  Scheduler scheduler = new Scheduler(
            configsBuilder.checkpointConfig(),
            configsBuilder.coordinatorConfig(),
            configsBuilder.leaseManagementConfig(),
            configsBuilder.lifecycleConfig(),
            configsBuilder.metricsConfig(),
            configsBuilder.processorConfig(),
            retrievalConfig
        );

 public void processRecords(ProcessRecordsInput processRecordsInput) {
            MDC.put(SHARD_ID_MDC_KEY, shardId);
            try {
                log.info("Processing {} record(s)",
                processRecordsInput.records().size());
                processRecordsInput.records()
                .forEach(
                    r ->
                        log.info("Processed record pk: {} -- Seq: {} : data {} with
 schema: {}",
                        r.partitionKey(),
 r.sequenceNumber(), recordToAvroObj(r).toString(), r.getSchema()));
            } catch (Throwable t) {
                log.error("Caught throwable while processing records. Aborting.");
                Runtime.getRuntime().halt(1);
```

```
            } finally {
                MDC.remove(SHARD_ID_MDC_KEY);
            }
    }

    private GenericRecord recordToAvroObj(KinesisClientRecord r) {
       byte[] data = new byte[r.data().remaining()];
       r.data().get(data, 0, data.length);
       org.apache.avro.Schema schema = new
    org.apache.avro.Schema.Parser().parse(r.schema().getSchemaDefinition());
       DatumReader datumReader = new GenericDatumReader<>(schema);

       BinaryDecoder binaryDecoder = DecoderFactory.get().binaryDecoder(data, 0,
    data.length, null);
       return (GenericRecord) datumReader.read(null, binaryDecoder);
    }
```

Interaktion mit Daten mithilfe der Kinesis Data Streams APIs

In diesem Abschnitt wird die Integration von Kinesis Data Streams mit Schema Registry mithilfe der Kinesis Data Streams beschrieben. APIs

1. Aktualisieren Sie diese Maven-Abhängigkeiten:

```
<dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>com.amazonaws</groupId>
                <artifactId>aws-java-sdk-bom</artifactId>
                <version>1.11.884</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>

    <dependencies>
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-kinesis</artifactId>
        </dependency>
```

```
        <dependency>
            <groupId>software.amazon.glue</groupId>
            <artifactId>schema-registry-serde</artifactId>
            <version>1.1.5</version>
        </dependency>

        <dependency>
            <groupId>com.fasterxml.jackson.dataformat</groupId>
            <artifactId>jackson-dataformat-cbor</artifactId>
            <version>2.11.3</version>
        </dependency>
    </dependencies>
```

2. Fügen Sie im Produzenten Schema-Header-Informationen mithilfe der `PutRecords`- oder `PutRecord`-API in Kinesis Data Streams hinzu.

```
//The following lines add a Schema Header to the record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
 DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
 GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
 GlueSchemaRegistryConfiguration(getConfigs()));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema,
 recordAsBytes);
```

3. Verwenden Sie im Produzenten die `PutRecords`- oder `PutRecord`-API, um den Datensatz in den Datenstrom einzufügen.
4. Entfernen Sie im Verbraucher den Schemadatensatz aus dem Header und serialisieren Sie einen Avro-Schemadatensatz.

```
//The following lines remove Schema Header from record
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
 GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
 getConfigs());
        byte[] recordWithSchemaHeaderBytes = new
 byte[recordWithSchemaHeader.remaining()];
```

```
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
recordWithSchemaHeaderBytes.length);
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);
        byte[] record =
glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);

        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }
```

Interaktion mit Daten mithilfe der Kinesis Data Streams APIs

Im Folgenden finden Sie einen Beispielcode für die Verwendung von `PutRecords` und `GetRecords` APIs.

```
//Full sample code
import
 com.amazonaws.services.schemaregistry.deserializers.GlueSchemaRegistryDeserializerImpl;
import
 com.amazonaws.services.schemaregistry.serializers.GlueSchemaRegistrySerializerImpl;
import com.amazonaws.services.schemaregistry.utils.AVROUtils;
import com.amazonaws.services.schemaregistry.utils.AWSSchemaRegistryConstants;
import org.apache.avro.Schema;
import org.apache.avro.generic.GenericData;
import org.apache.avro.generic.GenericDatumReader;
import org.apache.avro.generic.GenericDatumWriter;
import org.apache.avro.generic.GenericRecord;
import org.apache.avro.io.Decoder;
import org.apache.avro.io.DecoderFactory;
import org.apache.avro.io.Encoder;
import org.apache.avro.io.EncoderFactory;
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.services.glue.model.DataFormat;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
```

```
import java.nio.ByteBuffer;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;


public class PutAndGetExampleWithEncodedData {
    static final String regionName = "us-east-2";
    static final String streamName = "testStream1";
    static final String schemaName = "User-Topic";
    static final String AVRO_USER_SCHEMA_FILE = "src/main/resources/user.avsc";
    KinesisApi kinesisApi = new KinesisApi();

    void runSampleForPutRecord() throws IOException {
        Object testRecord = getTestRecord();
        byte[] recordAsBytes = convertRecordToBytes(testRecord);
        String schemaDefinition =
 AVROUtils.getInstance().getSchemaDefinition(testRecord);

        //The following lines add a Schema Header to a record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
 DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
 GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
 GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema, recordAsBytes);

        //Use PutRecords api to pass a list of records
        kinesisApi.putRecords(Collections.singletonList(recordWithSchemaHeader),
 streamName, regionName);

        //OR
        //Use PutRecord api to pass single record
        //kinesisApi.putRecord(recordWithSchemaHeader, streamName, regionName);
    }

    byte[] runSampleForGetRecord() throws IOException {
        ByteBuffer recordWithSchemaHeader = kinesisApi.getRecords(streamName,
 regionName);
```

```
        //The following lines remove the schema registry header
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(), new
GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeaderBytes = new
byte[recordWithSchemaHeader.remaining()];
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
recordWithSchemaHeaderBytes.length);

        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);

        byte[] record =
glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);

        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }

        return record;
    }

    private byte[] convertRecordToBytes(final Object record) throws IOException {
        ByteArrayOutputStream recordAsBytes = new ByteArrayOutputStream();
        Encoder encoder = EncoderFactory.get().directBinaryEncoder(recordAsBytes,
null);
        GenericDatumWriter datumWriter = new
GenericDatumWriter<>(AVROUtils.getInstance().getSchema(record));
        datumWriter.write(record, encoder);
        encoder.flush();
        return recordAsBytes.toByteArray();
    }

    private GenericRecord convertBytesToRecord(Schema avroSchema, byte[] record) throws
IOException {
        final GenericDatumReader<GenericRecord> datumReader = new
GenericDatumReader<>(avroSchema);
        Decoder decoder = DecoderFactory.get().binaryDecoder(record, null);
        GenericRecord genericRecord = datumReader.read(null, decoder);
```

```
        return genericRecord;
    }

    private Map<String, String> getMetadata() {
        Map<String, String> metadata = new HashMap<>();
        metadata.put("event-source-1", "topic1");
        metadata.put("event-source-2", "topic2");
        metadata.put("event-source-3", "topic3");
        metadata.put("event-source-4", "topic4");
        metadata.put("event-source-5", "topic5");
        return metadata;
    }

    private GlueSchemaRegistryConfiguration getConfigs() {
        GlueSchemaRegistryConfiguration configs = new
 GlueSchemaRegistryConfiguration(regionName);
        configs.setSchemaName(schemaName);
        configs.setAutoRegistration(true);
        configs.setMetadata(getMetadata());
        return configs;
    }

    private Object getTestRecord() throws IOException {
        GenericRecord genericRecord;
        Schema.Parser parser = new Schema.Parser();
        Schema avroSchema = parser.parse(new File(AVRO_USER_SCHEMA_FILE));

        genericRecord = new GenericData.Record(avroSchema);
        genericRecord.put("name", "testName");
        genericRecord.put("favorite_number", 99);
        genericRecord.put("favorite_color", "red");

        return genericRecord;
    }
}
```

Anwendungsfall: Amazon Managed Service für Apache Flink

Apache Flink ist ein beliebtes Open-Source-Framework und eine verteilte Verarbeitungs-Engine für statusbehaftete Berechnungen über unbegrenzte und begrenzte Datenströme. Amazon Managed Service für Apache Flink ist ein vollständig verwalteter AWS Service, mit dem Sie Apache Flink-Anwendungen zur Verarbeitung von Streaming-Daten erstellen und verwalten können.

Open Source Apache Flink bietet eine Reihe von Quellen und Senken. Vordefinierte Datenquellen umfassen beispielsweise das Lesen von Dateien, Verzeichnissen und Sockets sowie das Aufnehmen von Daten aus Sammlungen und Iteratoren. Apache Flink DataStream Connectors bieten Code für Apache Flink als Schnittstelle zu verschiedenen Systemen von Drittanbietern wie Apache Kafka oder Kinesis als Quellen und/oder Senken.

Weitere Informationen finden Sie im Amazon Kinesis Data Analytics-Entwicklerhandbuch.

Apache Flink Kafka Connector

Apache Flink bietet einen Apache-Kafka-Datenstrom-Konnektor für das Lesen von Daten aus Kafka-Themen und das Schreiben von Daten in Kafka-Themen mit Genau-Einmal-Garantie. Der Kafka-Verbraucher von Fink, `FlinkKafkaConsumer`, bietet Zugriff auf das Lesen aus einem oder mehreren Kafka-Themen. Der Apache-Flink-Kafka-Produzent `FlinkKafkaProducer` ermöglicht das Schreiben eines Streams von Datensätzen zu einem oder mehreren Kafka-Themen. Weitere Informationen finden Sie unter Kinesis Kafka Konnektor.

Apache Flink Kinesis Streams Connector

Der Kinesis Data Stream Connector bietet Zugriff auf Amazon Kinesis Data Streams. Die `FlinkKinesisConsumer` ist eine exakt einmal parallel Streaming-Datenquelle, die mehrere Kinesis-Streams innerhalb derselben AWS Serviceregion abonniert und das Re-Sharding von Streams transparent handhaben kann, während der Job ausgeführt wird. Jede Unteraufgabe des Verbrauchers ist für das Abrufen von Datensätzen aus mehreren Kinesis-Shards verantwortlich. Die Anzahl der Shards, die von jeder Unteraufgabe abgerufen werden, ändert sich, wenn Shards geschlossen und von Kinesis erstellt werden. Der `FlinkKinesisProducer` verwendet die Kinesis Producer Library (KPL), um Daten aus einem Apache-Flink-Stream in einen Kinesis-Stream zu übertragen. Weitere Informationen finden Sie unter Amazon Kinesis Streams Connector.

Weitere Informationen finden Sie hier: AWS GlueSchema Github-Repository.

Integration mit Apache Flink

Die mit Schema Registry bereitgestellte SerDes Bibliothek ist in Apache Flink integriert. Um mit Apache Flink zu arbeiten, müssen Sie die Schnittstellen `SerializationSchema` und `DeserializationSchema` namens `GlueSchemaRegistryAvroSerializationSchema` und `GlueSchemaRegistryAvroDeserializationSchema` implementieren, die Sie in Apache-Flink-Konnektoren einbinden können.

Hinzufügen eines AWS Glue Abhängigkeit von der Schemaregistrierung zur Apache Flink-Anwendung

Um die Integrationsabhängigkeiten einzurichten für AWS Glue Schemaregistrierung in der Apache Flink-Anwendung:

1. Fügen Sie die Abhängigkeit zur `pom.xml`-Datei hinzu.

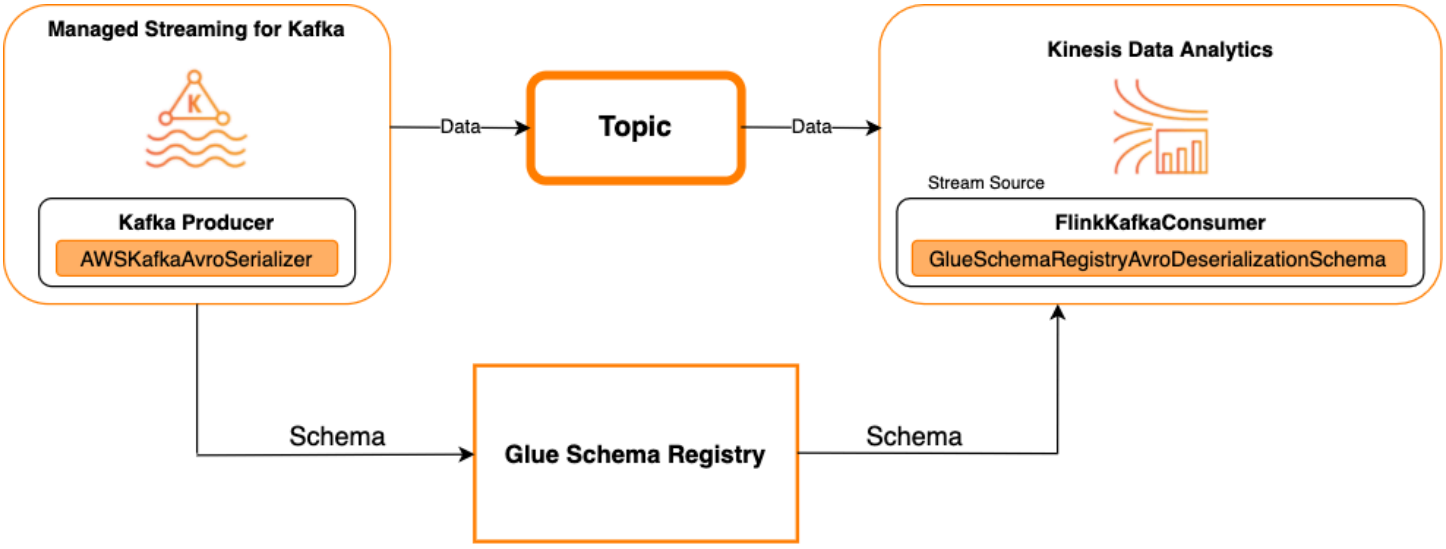
```
<dependency>
    <groupId>software.amazon.glue</groupId>
    <artifactId>schema-registry-flink-serde</artifactId>
    <version>1.0.0</version>
</dependency>
```

Integration von Kafka oder Amazon MSK mit Apache Flink

Sie können Managed Service für Apache Flink für Apache Flink mit Kafka als Quelle oder Kafka als Senke verwenden.

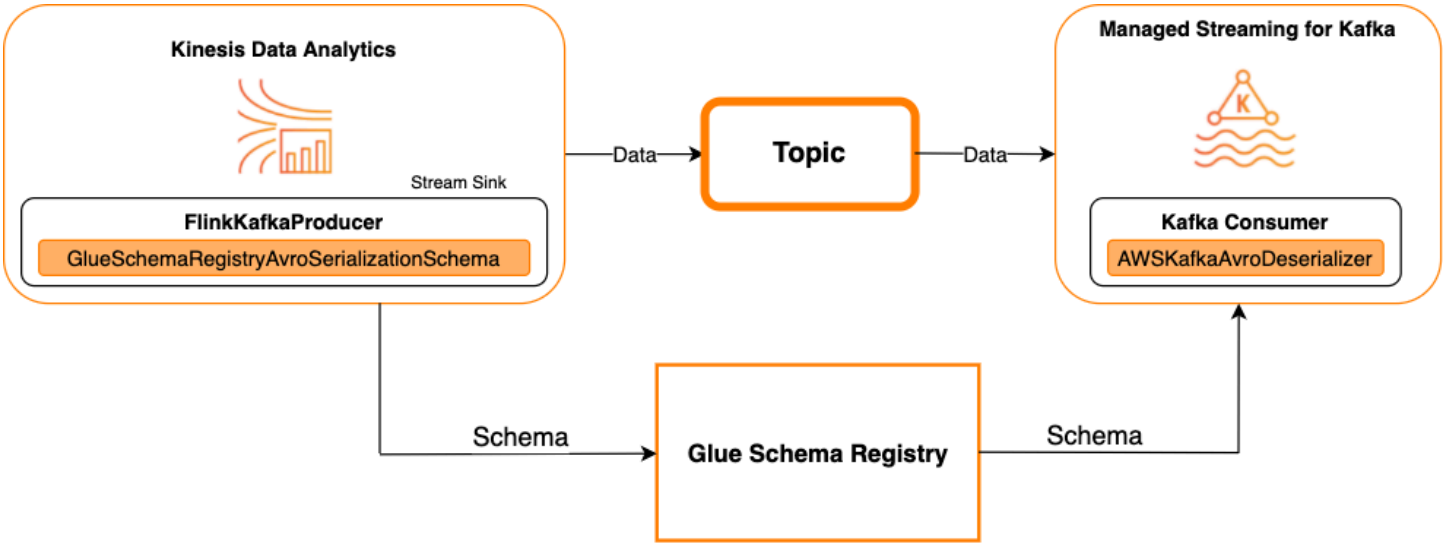
Kafka als Quelle

Das folgende Diagramm zeigt die Integration von Kinesis Data Streams mit Managed Service für Apache Flink für Apache Flink, mit Kafka als Quelle.



Kafka als Senke

Das folgende Diagramm zeigt die Integration von Kinesis Data Streams mit Managed Service für Apache Flink für Apache Flink, mit Kafka als Senke.



Um Kafka (oder Amazon MSK) in Managed Service für Apache Flink für Apache Flink zu integrieren, mit Kafka als Quelle oder Kafka als Senke, nehmen Sie die folgenden Codeänderungen vor. Fügen Sie in den entsprechenden Abschnitten die fett formatierten Codeblöcke zu Ihrem jeweiligen Code hinzu.

Wenn Kafka die Quelle ist, verwenden Sie den Deserialisierer-Code (Block 2). Wenn Kafka die Senke ist, verwenden Sie den Serialisierer-Code (Block 3).

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

String topic = "topic";
Properties properties = new Properties();
properties.setProperty("bootstrap.servers", "localhost:9092");
properties.setProperty("group.id", "test");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKafkaConsumer<GenericRecord> consumer = new FlinkKafkaConsumer<>(
    topic,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
    properties);
```

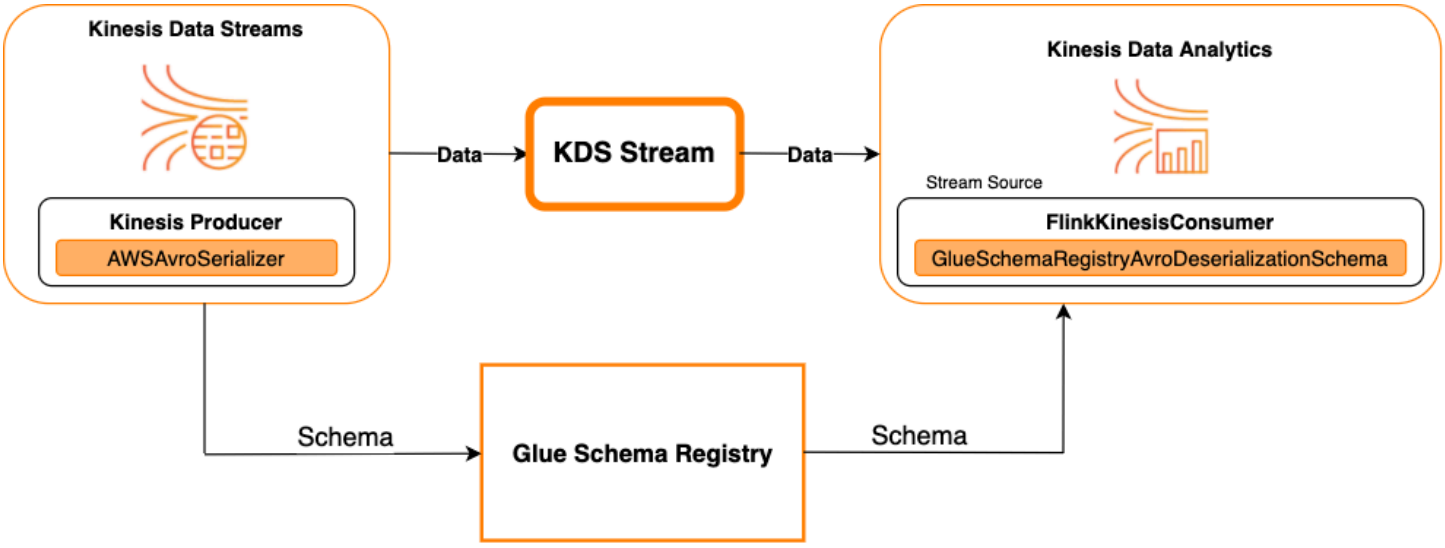
```
FlinkKafkaProducer<GenericRecord> producer = new FlinkKafkaProducer<>(
    topic,
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Integrieren von Kinesis Data Streams in Apache Flink

Sie können Managed Service für Apache Flink für Apache Flink mit Kinesis Data Streams als Quelle oder Senke verwenden.

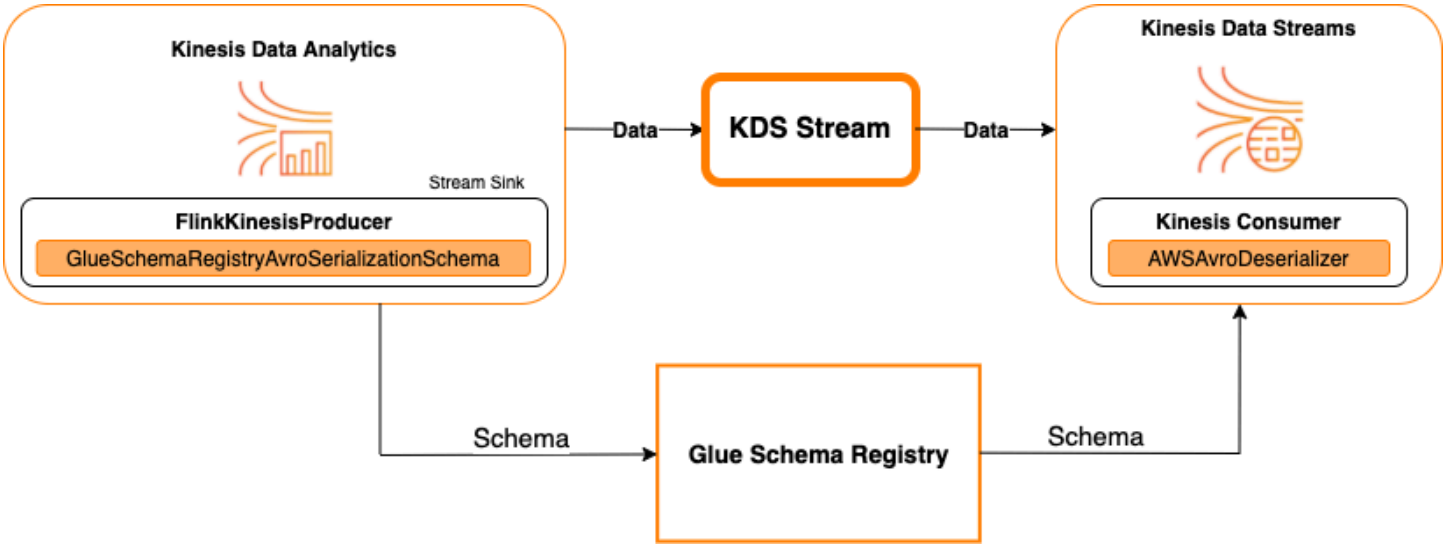
Kinesis Data Streams als Quelle

Das folgende Diagramm zeigt die Integration von Kinesis Data Streams mit Managed Service für Apache Flink für Apache Flink, mit Kinesis Data Streams als Quelle.



Kinesis Data Streams als Senke

Das folgende Diagramm zeigt die Integration von Kinesis Data Streams mit Managed Service für Apache Flink für Apache Flink, mit Kinesis Data Streams als Senke.



Um Kinesis Data Streams mit Managed Service für Apache Flink für Apache Flink zu integrieren, mit Kinesis Data Streams als Quelle oder Kinesis Data Streams als Senke, nehmen Sie die folgenden Codeänderungen vor. Fügen Sie in den entsprechenden Abschnitten die fett formatierten Codeblöcke zu Ihrem jeweiligen Code hinzu.

Wenn Kinesis Data Streams die Quelle ist, verwenden Sie den Deserialisierer-Code (Block 2). Wenn Kinesis Data Streams die Senke ist, verwenden Sie den Serialisierer-Code (Block 3).

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

String streamName = "stream";
Properties consumerConfig = new Properties();
consumerConfig.put(AWSConfigConstants.AWS_REGION, "aws-region");
consumerConfig.put(AWSConfigConstants.AWS_ACCESS_KEY_ID, "aws_access_key_id");
consumerConfig.put(AWSConfigConstants.AWS_SECRET_ACCESS_KEY, "aws_secret_access_key");
consumerConfig.put(ConsumerConfigConstants.STREAM_INITIAL_POSITION, "LATEST");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKinesisConsumer<GenericRecord> consumer = new FlinkKinesisConsumer<>(
    streamName,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
```

```
    properties);

FlinkKinesisProducer<GenericRecord> producer = new FlinkKinesisProducer<>(
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);
producer.setDefaultStream(streamName);
producer.setDefaultPartition("0");

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Anwendungsfall: Integration mit AWS Lambda

Um eine AWS Lambda Funktion als Apache Kafka/Amazon MSK-Consumer zu verwenden und AVRO-kodierte Nachrichten zu deserialisieren mit AWS Glue Schema Registry finden Sie auf der MSK Labs-Seite.

Anwendungsfall: AWS Glue Data Catalog

AWS Glue Tabellen unterstützen Schemas, die Sie manuell oder durch Verweis auf Folgendes angeben können AWS Glue Schemaregistrierung. Die Schemaregistrierung ist in den Datenkatalog integriert, sodass Sie bei der Erstellung oder Aktualisierung optional in der Schemaregistry gespeicherte Schemas verwenden können AWS Glue Tabellen oder Partitionen im Datenkatalog. Um eine Schemadefinition in der Schema Registry zu identifizieren, müssen Sie mindestens den ARN des Schemas kennen, zu dem diese gehört. Eine Schemaversion eines Schemas, die eine Schemadefinition enthält, kann durch seine UUID oder Versionsnummer referenziert werden. Es gibt immer eine Schemaversion, die „neueste“ Version, die nachgeschlagen werden kann, ohne die Versionsnummer oder UUID zu kennen.

Bei Aufrufen der `CreateTable`- oder `UpdateTable`-Operationen übergeben Sie eine `TableInput`-Struktur mit einem `StorageDescriptor`, der unter Umständen eine `SchemaReference` auf ein vorhandenes Schema in der Schema Registry hat. In ähnlicher Weise kann die Antwort beim Aufrufen von `GetTable` oder `GetPartition` APIs das Schema und das enthalten`SchemaReference`. Wenn eine Tabelle oder Partition mit Schemareferenzen erstellt wurde, versucht der Data Catalog, das Schema für diese Schemareferenz abzurufen. Falls er das Schema in der Schema Registry nicht findet, wird ein leeres Schema in der `GetTable`-Antwort zurückgegeben. Andernfalls enthält die Antwort sowohl das Schema als auch die Schemareferenz.

Sie können die Aktionen auch von der AWS Glue console.

Um diese Vorgänge durchzuführen und die Schemainformationen zu erstellen, zu aktualisieren oder anzuzeigen, müssen Sie dem aufrufenden Benutzer eine IAM-Rolle zuweisen, die Berechtigungen für die `GetSchemaVersion`-API bereitstellt.

Hinzufügen einer Tabelle oder Aktualisieren des Schemas für eine Tabelle

Das Hinzufügen einer neuen Tabelle aus einem vorhandenen Schema bindet die Tabelle an eine bestimmte Schemaversion. Sobald neue Schemaversionen registriert wurden, können Sie diese Tabellendefinition auf der Seite Tabelle anzeigen im AWS Glue Konsole oder mithilfe der UpdateTable Aktion (Python: update_table) API.

Hinzufügen einer Tabelle aus einem vorhandenen Schema

Sie können eine erstellen AWS Glue Tabelle aus einer Schemaversion in der Registrierung mit dem AWS Glue Konsole oder `CreateTable` API.

AWS Glue API

Bei Aufrufen der `CreateTable`-API übergeben Sie eine `TableInput` mit einem `StorageDescriptor` und einer `SchemaReference` zu einem vorhandenen Schema in der Schema Registry.

AWS Glue Konsole

Um eine Tabelle aus dem zu erstellen AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter Data Catalog die Option Tables (Tabellen).
3. Wählen Sie im Menü Add Tables (Tabellen hinzufügen) die Option Add table from existing schema (Tabelle aus vorhandenem Schema hinzufügen).
4. Konfigurieren Sie die Tabelleneigenschaften und den Datenspeicher gemäß AWS Glue Entwicklerhandbuch.
5. Wählen Sie auf der Seite Choose a Glue schema (Glue-Schema wählen) die Registry, in dem sich das Schema befindet.
6. Wählen Sie den Schema name (Schemaname) und wählen Sie die Version des anzuwendenden Schemas.

7. Überprüfen Sie die Schemavorschau und klicken Sie auf Next (Weiter).
8. Überprüfen und erstellen Sie die Tabelle.

Das Schema und die Version, die auf die Tabelle angewendet werden, werden in der Spalte Glue-Schema in der Liste der Tabellen angezeigt. Sie können die Tabelle anzeigen, um weitere Details zu sehen.

Aktualisieren des Schemas für eine Tabelle

Wenn eine neue Schemaversion verfügbar ist, möchten Sie möglicherweise das Schema einer Tabelle mithilfe der UpdateTable Aktion (Python: update_table) API oder der aktualisieren AWS Glue console.

> **Important**
>
> Wenn Sie das Schema für eine vorhandene Tabelle aktualisieren, die über AWS Glue Das manuell angegebene Schema ist möglicherweise nicht kompatibel mit dem neuen Schema, auf das in der Schemaregistrierung verwiesen wird. Dies kann dazu führen, dass Ihre Aufträge fehlschlagen.

AWS Glue API

Bei Aufrufen der `UpdateTable`-API übergeben Sie eine `TableInput` mit einem `StorageDescriptor` und einer `SchemaReference` zu einem vorhandenen Schema in der Schema Registry.

AWS Glue Konsole

Um das Schema für eine Tabelle aus dem zu aktualisieren AWS Glue Konsole:

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter Data Catalog die Option Tables (Tabellen).
3. Zeigen Sie die Tabelle aus der Liste der Tabellen an.
4. Klicken Sie auf Update schema (Schema aktualisieren) in dem Feld, das Sie über die neue Version informiert.

5. Überprüfen Sie die Unterschiede zwischen dem aktuellen und dem neuen Schema.
6. Klicken Sie auf Show all schema differences (Alle Schemaunterschiede anzeigen), um weitere Details zu sehen.
7. Klicken Sie auf Save table (Tabelle speichern), um die neue Version zu akzeptieren.

Anwendungsfall: AWS Glue Streamen

AWS Glue Beim Streaming werden Daten aus Streaming-Quellen verbraucht und ETL-Operationen ausgeführt, bevor sie in eine Ausgabesenke geschrieben werden. Die Eingabe-Streaming-Quelle kann mit einer Datentabelle oder direkt durch Angabe der Quellkonfiguration angegeben werden.

AWS Glue Streaming unterstützt eine Datenkatalogtabelle für die Streaming-Quelle, die mit dem Schema erstellt wurde, das in AWS Glue Schemaregistrierung. Sie können ein Schema erstellen in AWS Glue Schemaregistrierung und erstellen Sie ein AWS Glue Tabelle mit einer Streaming-Quelle, die dieses Schema verwendet. Dieser AWS Glue Tabelle kann als Eingabe für eine verwendet werden AWS Glue Streaming-Job zum Deserialisieren von Daten im Eingabestream.

Ein Punkt, der hier zu beachten ist, ist, wenn das Schema in der AWS Glue Wenn sich die Schemaregistrierung ändert, müssen Sie das neu starten AWS Glue Der Streaming-Job muss die Änderungen im Schema widerspiegeln.

Anwendungsfall: Apache Kafka Streams

Die Apache Kafka Streams API ist eine Client-Bibliothek zur Verarbeitung und Analyse von Daten, die in Apache Kafka gespeichert sind. Dieser Abschnitt beschreibt die Integration von Apache Kafka Streams mit AWS Glue Schema Registry, mit der Sie Schemas in Ihren Datenstreaming-Anwendungen verwalten und durchsetzen können. Weitere Informationen zu Apache Kafka Streams finden Sie unter Apache Kafka Streams.

Integration mit den Bibliotheken SerDes

Es gibt eine `GlueSchemaRegistryKafkaStreamsSerde`-Klasse, mit der Sie eine Streams-Anwendung konfigurieren können.

Beispielcode für die Kafka-Streams-Anwendung

Um das AWS Glue Schemaregistrierung innerhalb einer Apache Kafka Streams-Anwendung:

1. Konfigurieren Sie die Kafka-Streams-Anwendung.

```
final Properties props = new Properties();
    props.put(StreamsConfig.APPLICATION_ID_CONFIG, "avro-streams");
    props.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
    props.put(StreamsConfig.CACHE_MAX_BYTES_BUFFERING_CONFIG, 0);
    props.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG,
 Serdes.String().getClass().getName());
    props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG,
 AWSKafkaAvroSerDe.class.getName());
    props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");

    props.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
    props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
    props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());
 props.put(AWSSchemaRegistryConstants.DATA_FORMAT, DataFormat.AVRO.name());
```

2. Erstellen Sie einen Stream aus dem Thema avro-input.

```
StreamsBuilder builder = new StreamsBuilder();
final KStream<String, GenericRecord> source = builder.stream("avro-input");
```

3. Verarbeiten Sie die Datensätze (das Beispiel filtert die Datensätze heraus, deren Wert favorite_color pink ist oder bei denen der Wert von „amount“ 15 ist).

```
final KStream<String, GenericRecord> result = source
    .filter((key, value) -
> !"pink".equals(String.valueOf(value.get("favorite_color"))));
    .filter((key, value) -> !"15.0".equals(String.valueOf(value.get("amount"))));
```

4. Schreiben Sie die Ergebnisse zurück in das Thema avro-output.

```
result.to("avro-output");
```

5. Starten Sie die Apache-Kafka-Streams-Anwendung.

```
KafkaStreams streams = new KafkaStreams(builder.build(), props);
streams.start();
```

Ergebnisse der Implementierung

Diese Ergebnisse zeigen den Filtervorgang von Datensätzen, die in Schritt 3 als favorite_color mit „pink“ oder „15.0“ herausgefiltert wurden.

Datensätze vor dem Filtern:

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}
{"name": "Jay", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
{"id": "entertainment_2","amount": 105}
 {"id": "commute_1","amount": 15}
```

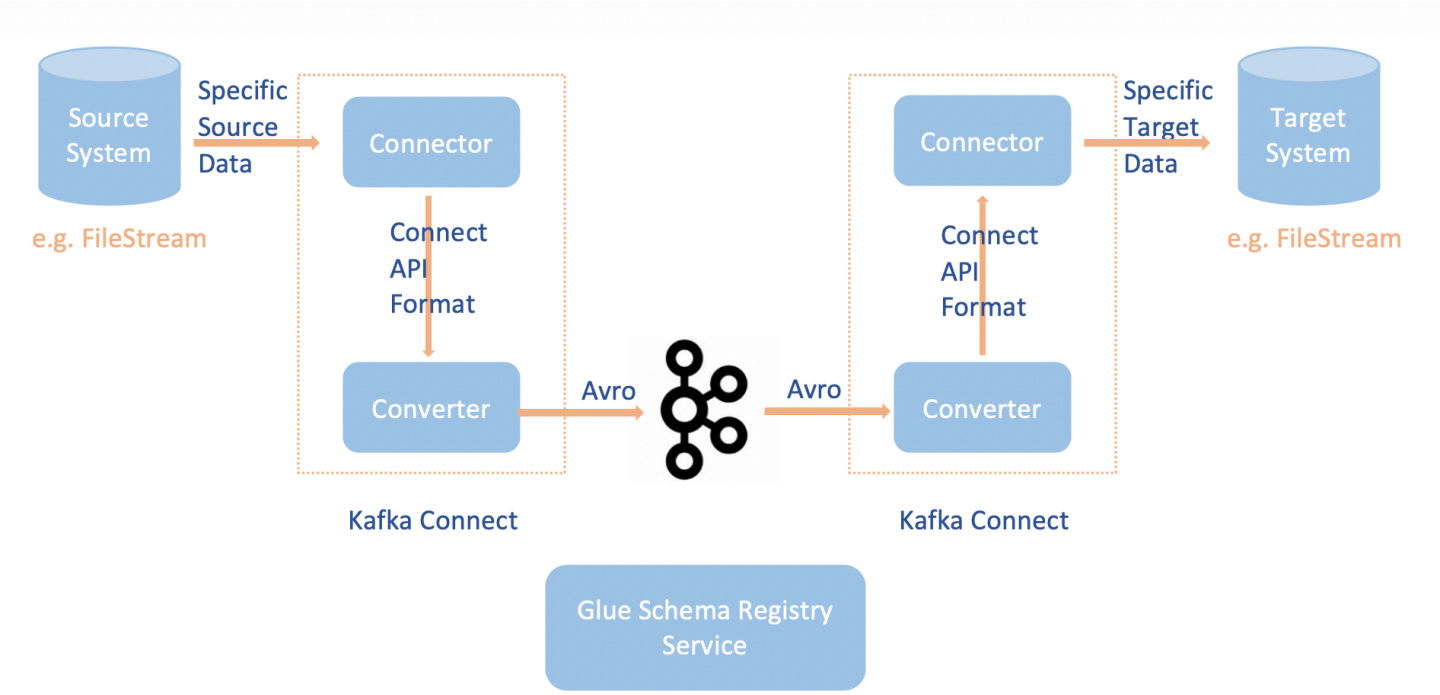
Datensätze nach dem Filtern:

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
{"id": "entertainment_2","amount": 105}
```

Anwendungsfall: Apache Kafka Connect

Die Integration von Apache Kafka Connect mit dem AWS Glue Mit Schema Registry können Sie Schemainformationen von Konnektoren abrufen. Die Apache-Kafka-Konverter geben das Format der Daten in Apache Kafka an, und wie diese in Apache-Kafka-Connect-Daten übersetzt werden. Jeder

Apache-Kafka-Connect-Benutzer muss diese Konverter basierend auf dem Format konfigurieren, in dem seine Daten geladen oder in Apache Kafka gespeichert werden sollen. Auf diese Weise können Sie Ihre eigenen Konverter definieren, um Apache Kafka Connect-Daten in den Typ zu übersetzen, der in der AWS Glue Schema Registry (zum Beispiel: Avro) und unseren Serializer verwenden, um das Schema zu registrieren und die Serialisierung durchzuführen. Dann können Konverter auch unseren Deserializer verwenden, um die von Apache Kafka empfangenen Daten zu deserialisieren und wieder in Apache-Kafka-Connect-Daten zu konvertieren. Ein Beispiel für ein Workflow-Diagramm ist unten angegeben.



1. Installieren Sie das `aws-glue-schema-registry` Projekt, indem Sie das Github-Repository für klonen AWS Glue Schema-Registrierung.

```
git clone git@github.com:awslabs/aws-glue-schema-registry.git
cd aws-glue-schema-registry
mvn clean install
mvn dependency:copy-dependencies
```

2. Wenn Sie planen, Apache Kafka Connect im Standalone-Modus zu verwenden, aktualisieren Sie die connect-standalone.properties mit der untenstehenden Anleitung für diesen Schritt. Wenn Sie Apache Kafka Connect im verteilten Modus verwenden möchten, aktualisieren Sie connect-avro-distributed.properties mit denselben Anweisungen.

   a. Fügen Sie diese Eigenschaften auch der Apache-Kafka-Connect-Properties-Datei hinzu:

```
key.converter.region=aws-region
value.converter.region=aws-region
key.converter.schemaAutoRegistrationEnabled=true
value.converter.schemaAutoRegistrationEnabled=true
key.converter.avroRecordType=GENERIC_RECORD
value.converter.avroRecordType=GENERIC_RECORD
```

b. Fügen Sie den folgenden Befehl zum Abschnitt Startmodus unter kafka-run-class .sh hinzu:

```
-cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
```

3. Fügen Sie den folgenden Befehl zum Abschnitt Startmodus unter kafka-run-class .sh hinzu

```
-cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
```

Das sollte wie folgt aussehen:

```
# Launch mode
if [ "x$DAEMON_MODE" = "xtrue" ]; then
  nohup "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
 $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
registry/target/dependency/*" $KAFKA_OPTS "$@" > "$CONSOLE_OUTPUT_FILE" 2>&1 < /dev/
null &
else
  exec "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
 $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
registry/target/dependency/*" $KAFKA_OPTS "$@"
fi
```

4. Wenn Sie bash verwenden, führen Sie die folgenden Befehle aus, um Ihren CLASSPATH in Ihrem bash_profile einzurichten. Aktualisieren Sie die Umgebung für jede andere Shell entsprechend.

```
echo 'export GSR_LIB_BASE_DIR=<>' >>~/.bash_profile
echo 'export GSR_LIB_VERSION=1.0.0' >>~/.bash_profile
echo 'export KAFKA_HOME=<your Apache Kafka installation directory>' >>~/.bash_profile
echo 'export CLASSPATH=$CLASSPATH:$GSR_LIB_BASE_DIR/avro-kafkaconnect-converter/
target/schema-registry-kafkaconnect-converter-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
common/target/schema-registry-common-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
avro-serializer-deserializer/target/schema-registry-serde-$GSR_LIB_VERSION.jar'
 >>~/.bash_profile
```

```
source ~/.bash_profile
```

5. (Optional) Wenn Sie mit einer einfachen Dateiquelle testen möchten, klonen Sie den Dateiquellen-Konnektor.

```
git clone https://github.com/mmolimar/kafka-connect-fs.git
cd kafka-connect-fs/
```

   a. Ändern Sie unter der Konfiguration des Quellen-Konnektors das Datenformat auf Avro, den Datei-Reader auf`AvroFileReader` und aktualisieren Sie ein Beispiel-Avro-Objekt aus dem Dateipfad, aus dem Sie lesen. Zum Beispiel:

```
vim config/kafka-connect-fs.properties
```

```
fs.uris=<path to a sample avro object>
policy.regexp=^.*\.avro$
file_reader.class=com.github.mmolimar.kafka.connect.fs.file.reader.AvroFileReader
```

   b. Installieren Sie den Quellen-Konnektor.

```
mvn clean package
echo "export CLASSPATH=\$CLASSPATH:\"\$(find target/ -type f -name '*.jar'| grep
 '\-package' | tr '\n' ':')\"" >>~/.bash_profile
source ~/.bash_profile
```

   c. Aktualisieren Sie die Senkeneigenschaften unter *<your Apache Kafka installation directory>*`/config/connect-file-sink.properties`, aktualisieren Sie den Namen des Themas und den Dateinamen.

```
file=<output file full path>
topics=<my topic>
```

6. Starten Sie den Quellen-Konnektor (in diesem Beispiel handelt es sich um einen Dateiquellen-Konnektor).

```
$KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
standalone.properties config/kafka-connect-fs.properties
```

7. Führen Sie den Quellen-Konnektor aus (in diesem Beispiel handelt es sich um einen Dateiquellen-Konnektor).

```
$KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
standalone.properties $KAFKA_HOME/config/connect-file-sink.properties
```

Ein Beispiel für die Verwendung von Kafka Connect finden Sie im Skript run-local-tests .sh im Ordner integration-tests im Github-Repository für AWS GlueSchemaregistrierung.

Migration von einer Schemaregistrierung eines Drittanbieters zu AWS Glue Schema Registry

Die Migration von einer Schemaregistrierung eines Drittanbieters zur AWS Glue Schema Registry ist von der bestehenden, aktuellen Schemaregistry eines Drittanbieters abhängig. Wenn in einem Apache-Kafka-Thema Datensätze vorhanden sind, die mithilfe einer Drittanbieter-Schemaregistrierung gesendet wurden, benötigen Verbraucher diese Schemaregistrierung, um die Datensätze zu deserialisieren. Der `AWSKafkaAvroDeserializer` bietet die Möglichkeit, eine sekundäre Deserializer-Klasse anzugeben, die auf den Drittanbieter-Deserializer verweist und zum Deserialisieren dieser Datensätze verwendet wird.

Es gibt zwei Kriterien für das Ausmustern eines Drittanbieter-Schemas. Erstens kann das Ausmustern nur erfolgen, wenn Datensätze in Apache-Kafka-Themen, die die Drittanbieter-Schemaregistrierung verwenden, nicht mehr von und für Verbraucher benötigt werden. Zweitens kann der Ruhestand durch Alterung der Apache-Kafka-Themen erfolgen, abhängig von der für diese Themen festgelegten Aufbewahrungsfrist. Beachten Sie, dass Sie auch dann, wenn Sie Themen haben, deren Archivierung unbegrenzt ist, auf die AWS Glue Schemaregistrierung, aber Sie können die Schemaregistrierung eines Drittanbieters nicht außer Betrieb nehmen. Um das Problem zu umgehen, können Sie eine Anwendung oder Mirror Maker 2 verwenden, um aus dem aktuellen Thema vorzulesen und mit dem AWS Glue Schemaregistrierung.

Um von einer Schemaregistrierung eines Drittanbieters zur AWS Glue Schemaregistrierung:

1. Erstellen Sie eine Registrierung in der AWS Glue Schemaregistrierung, oder verwenden Sie die Standardregistrierung.
2. Stoppen Sie den Konsumenten. Ändern Sie es so, dass es einschließt AWS Glue Schema Registry als primärer Deserializer und die Schemaregistry eines Drittanbieters als sekundäre.
   - Legen Sie die Verbrauchereigenschaften fest. In diesem Beispiel wird der secondary_deserializer auf einen anderen Deserializer gesetzt. Das Verhalten ist wie folgt: Der Verbraucher ruft Datensätze aus Amazon MSK ab und versucht zunächst den

`AWSKafkaAvroDeserializer`. Wenn es nicht in der Lage ist, das magische Byte zu lesen, das die Avro-Schema-ID für AWS Glue Schemaregistrierungsschema, das versucht `AWSKafkaAvroDeserializer` dann, die im secondary_deserializer bereitgestellte Deserializer-Klasse zu verwenden. Die für den sekundären Deserializer spezifischen Eigenschaften müssen auch in den Verbrauchereigenschaften bereitgestellt werden, z. B. in den Eigenschaften schema_registry_url_config und specific_avro_reader_config, wie unten dargestellt.

```
consumerProps.setProperty(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
consumerProps.setProperty(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroDeserializer.class.getName());
consumerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION,
 KafkaClickstreamConsumer.gsrRegion);
consumerProps.setProperty(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 KafkaAvroDeserializer.class.getName());
consumerProps.setProperty(KafkaAvroDeserializerConfig.SCHEMA_REGISTRY_URL_CONFIG,
 "URL for third-party schema registry");
consumerProps.setProperty(KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG,
 "true");
```

3. Starten Sie den Verbraucher neu.
4. Stoppen Sie den Producer und verweisen Sie den Producer auf AWS Glue Schemaregistrierung.
   a. Legen Sie die Produzenteneigenschaften fest. In diesem Beispiel verwendet der Produzent die Schemaversionen default-registry und auto register.

```
producerProps.setProperty(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
 StringSerializer.class.getName());
producerProps.setProperty(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroSerializer.class.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
producerProps.setProperty(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.SPECIFIC_RECORD.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING,
 "true");
```

5. (Optional) Verschieben Sie vorhandene Schemas und Schemaversionen manuell aus der aktuellen Schemaregistrierung eines Drittanbieters in die AWS Glue Schemaregistrierung, entweder in die Standardregistrierung in AWS Glue Schemaregistrierung oder in eine bestimmte,

nicht standardmäßige Registrierung in AWS Glue Schemaregistrierung. Dies kann erreicht werden, indem Schemas aus den Schemaregistern von Drittanbietern im JSON-Format exportiert und neue Schemas erstellt werden in AWS Glue Schemaregistrierung mit dem oder dem AWS Management Console . AWS CLI

Dieser Schritt kann wichtig sein, wenn Sie Kompatibilitätsprüfungen mit früheren Schemaversionen für neu erstellte Schemaversionen mit AWS CLI und dem aktivieren müssen AWS Management Console, oder wenn Produzenten Nachrichten mit einem neuen Schema senden, bei dem die automatische Registrierung von Schemaversionen aktiviert ist.

6. Starten Sie den Produzenten.

Herstellen einer Verbindung zu Daten

Eine AWS Glue Verbindung ist ein Datenkatalogobjekt, das Anmeldeinformationen, URI-Zeichenfolgen, VPC-Informationen (Virtual Private Cloud) und mehr für einen bestimmten Datenspeicher speichert. AWS Glue Crawler, Jobs und Entwicklungsendpunkte verwenden Verbindungen, um auf bestimmte Arten von Datenspeichern zuzugreifen. Sie können Verbindungen sowohl für Quellen als auch für Ziele verwenden und dieselbe Verbindung für mehrere Crawler- oder ETL-Aufträge (Extract, Transform, Load) verwenden.

Die neueste Version des AWS Glue Verbindungsschemas bietet eine einheitliche Möglichkeit, Datenverbindungen zwischen AWS Diensten und Anwendungen wie AWS Glue Amazon Athena, und Amazon SageMaker AI Unified Studio zu verwalten.

Übersicht zur Verwendung von Connectors und Verbindungen

Eine Verbindung enthält die Eigenschaften, die für das Verbinden mit einem bestimmten Datenspeicher erforderlich sind. Wenn Sie eine Verbindung erstellen, wird sie im AWS Glue Data Catalog gespeichert. Sie wählen einen Connector aus und erstellen dann eine Verbindung basierend auf diesem Connector.

Sie können Konnektoren für Datenspeicher abonnieren, die nicht nativ unterstützt werden AWS Marketplace, und diese Konnektoren dann beim Erstellen von Verbindungen verwenden. Entwickler können auch eigene Connectors erstellen und sie beim Erstellen von Verbindungen verwenden.

> **Note**
>
> Verbindungen, die mit benutzerdefinierten Verbindungen oder AWS Marketplace Konnektoren in erstellt wurden, AWS Glue Studio werden in der AWS Glue Konsole mit dem Typ auf angezeigt. `UNKNOWN`

Die folgenden Schritte beschreiben den Gesamtprozess der Verwendung von Konnektoren in AWS Glue Studio:

1. Abonnieren Sie einen Connector in AWS Marketplace oder entwickeln Sie Ihren eigenen Connector und laden Sie ihn auf hoch AWS Glue Studio. Weitere Informationen finden Sie unter Konnektoren werden hinzugefügt zu AWS Glue Studio.

2. Überprüfen Sie die Nutzungsinformationen des Connectors. Sie finden diese Informationen auf der Registerkarte Usage (Verwendung) auf der Produktseite des Connectors. Wenn Sie beispielsweise auf dieser Produktseite, AWS Glue Connector für Google, auf den Tab Nutzung klicken BigQuery, finden Sie im Abschnitt Zusätzliche Ressourcen einen Link zu einem Blog über die Verwendung dieses Connectors.
3. Verbindung erstellen Sie wählen den Connector aus, den Sie verwenden möchten, und stellen zusätzliche Informationen für die Verbindung bereit, wie Anmeldeinformationen, URI-Zeichenfolgen und VPC-Informationen (Virtual Private Cloud). Weitere Informationen finden Sie unter Erstellen von Verbindungen für Connectors.
4. Erstellen Sie eine IAM-Rolle für Ihren Auftrag. Der Auftrag übernimmt die Berechtigungen der IAM-Rolle, die Sie angeben, wenn Sie sie erstellen. Diese IAM-Rolle muss über die nötigen Berechtigungen verfügen, sich bei Ihren Datenspeichern zu authentifizieren, Daten daraus zu extrahieren und darin zu schreiben.
5. Erstellen Sie einen ETL-Auftrag und konfigurieren Sie die Datenquelleneigenschaften für Ihren ETL-Auftrag. Geben Sie die Verbindungsoptionen und Authentifizierungsinformationen an, wie vom benutzerdefinierten Connector-Anbieter angewiesen. Weitere Informationen finden Sie unter Erstellen von Aufträgen mit benutzerdefinierten Connectors.
6. Passen Sie Ihren ETL-Auftrag an, indem Sie Transformationen oder zusätzliche Datenspeicher hinzufügen, wie unter Starten von Visual ETL-Jobs in AWS Glue Studio beschrieben.
7. Wenn Sie einen Connector für das Datenziel verwenden, konfigurieren Sie die Datenzieleigenschaften für Ihren ETL-Auftrag. Geben Sie die Verbindungsoptionen und Authentifizierungsinformationen an, wie vom benutzerdefinierten Connector-Anbieter angewiesen. Weitere Informationen finden Sie unter the section called “Erstellen von Aufträgen mit benutzerdefinierten Connectors”.
8. Passen Sie die Umgebung der Auftragsausführung an, indem Sie Auftragseigenschaften konfigurieren, wie unter Ändern der Auftragseigenschaften beschrieben.
9. Führen Sie den Auftrag aus.

Vereinheitlichte Verbindungen

AWS hat kürzlich eine neue Funktion namens „SageMaker LakeHouse Verbindungen“ oder „AWS Glue Unified Connections“ eingeführt. Mit dieser Funktion können Sie Verbindungen erstellen, die von mehreren AWS Diensten wie AWS Glue und verwendet werden können Amazon Athena. Wenn Sie eine Datenquelle in erstellen Amazon Athena, werden Sie einen Abschnitt sehen, der sich auf AWS Glue Verbindungseingaben bezieht. In diesem Fall Amazon Athena wird eine AWS

Glue Verbindung für Sie erstellt, einschließlich aller Amazon Athena-spezifischen Eigenschaften im Abschnitt AthenaProperties `` der Verbindung.

Wenn Sie dagegen direkt in eine Verbindung herstellen, werden Sie nur aufgefordert AWS Glue, spezifische Eigenschaften für Apache Spark einzugeben, die in den Abschnitten `` und ConnectionProperties `SparkProperties` der Verbindung gespeichert werden. AWS Glue

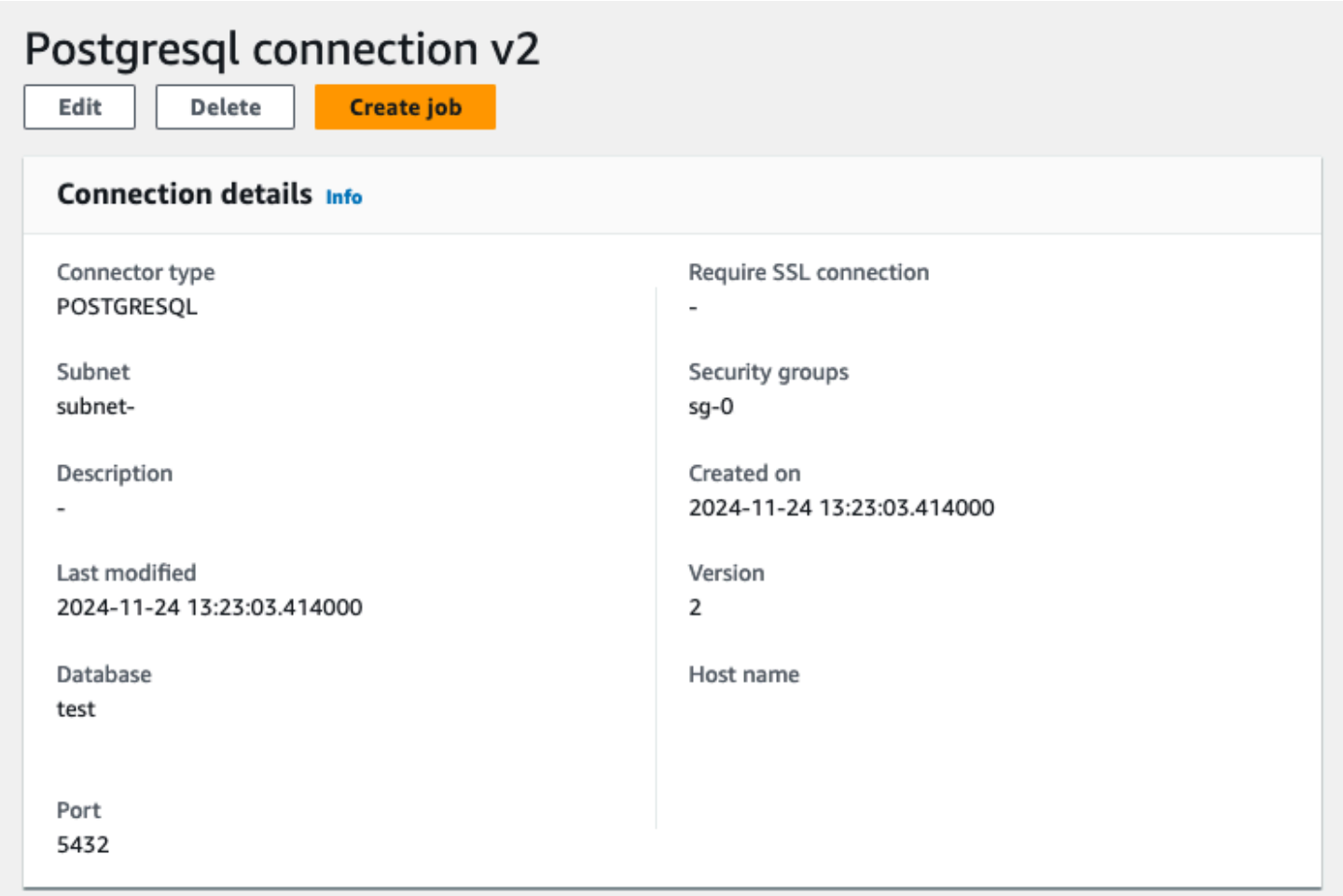
Beide Szenarien führen zur Erstellung einer „einheitlichen Verbindung“, aber die in erstellten Verbindungen Amazon Athena sind nur für die Verwendung innerhalb konfiguriert Amazon Athena, während die in erstellten Verbindungen nur für die Verwendung innerhalb konfiguriert AWS Glue sind AWS Glue. Es ist jedoch möglich, diese Verbindungen mit den fehlenden Eigenschaften (entweder Amazon Athena oder Spark-Eigenschaften) zu aktualisieren, sodass sie von beiden Diensten verwendet werden können. Amazon SageMaker AI Unified Studio kümmert sich automatisch darum, indem es alle erforderlichen Eigenschaften (``, ConnectionProperties `` und AthenaProperties `SparkProperties`) für die AWS Glue Verbindung ausfüllt AWS Glue und so sicherstellt, dass die Verbindung sowohl von als auch verwendet werden kann Amazon Athena.

Es ist wichtig zu beachten, dass wir diese Verbindungen zwar als „einheitliche Verbindungen“ bezeichnen, die in AWS Glue oder Amazon Athena einzeln erstellten Verbindungen jedoch nur dann wirklich vereinheitlicht sind, wenn sie für die Verwendung durch beide Dienste ordnungsgemäß konfiguriert sind. Nur die mit SageMaker Unified Studio erstellten Verbindungen sind wirklich vereinheitlicht und können sofort von mehreren Diensten verwendet werden.

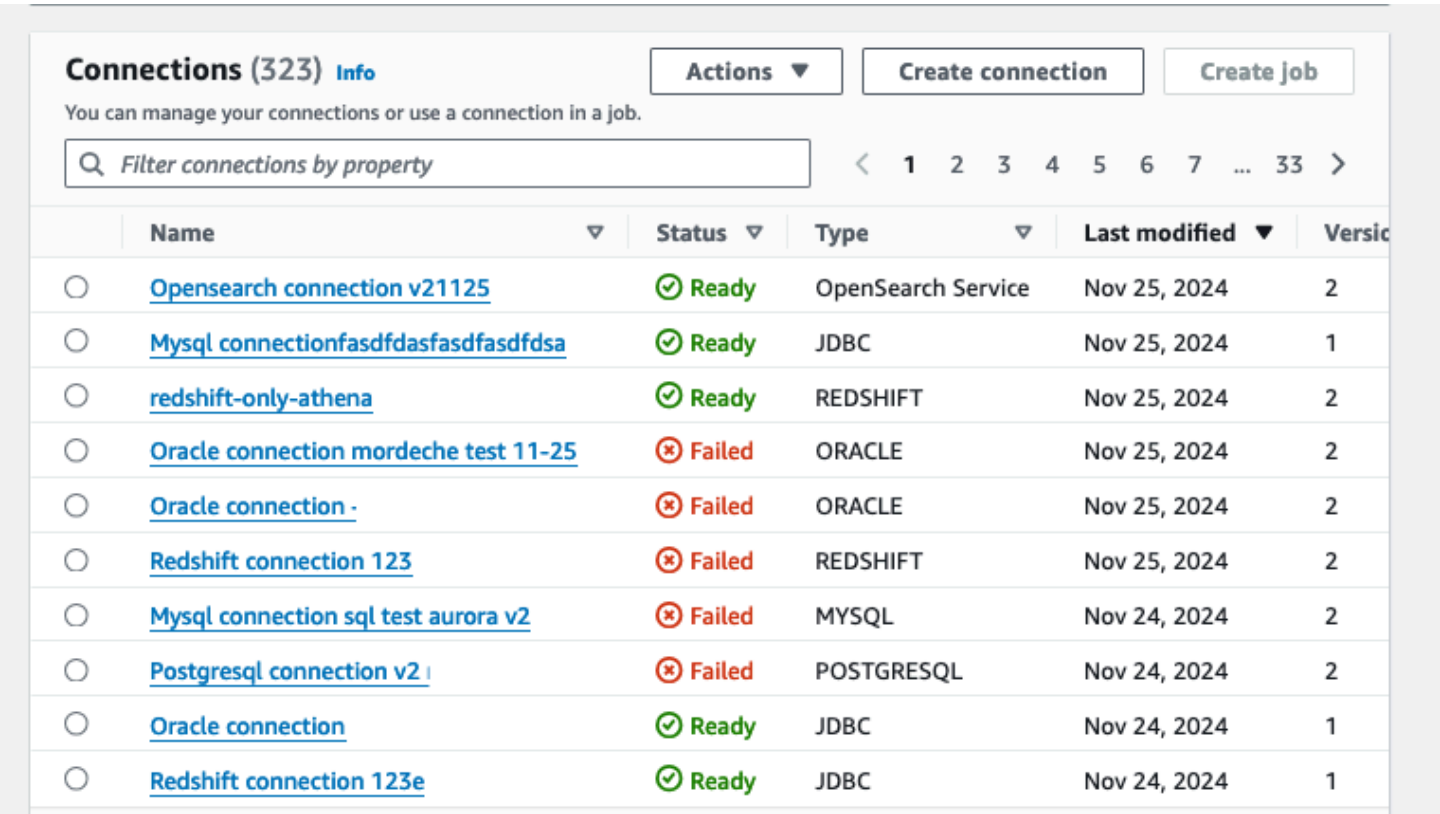
Darüber hinaus AWS Glue sind Verbindungen, die in erstellt wurden, nicht sichtbar, Amazon Athena da Datenquellen Amazon Athena angezeigt werden, die zwar einen Verweis auf eine AWS Glue Verbindung enthalten, aber nicht die AWS Glue Verbindung selbst sind. Ebenso Amazon Athena sind Verbindungen, die in erstellt wurden, nicht sichtbar, AWS Glue Studio da alle Verbindungen AWS Glue Studio herausgefiltert werden, für die nicht die erforderlichen Einstellungen konfiguriert wurden AWS Glue.

AWS Glue Studio erstellt standardmäßig einheitliche Verbindungen. In der AWS Glue Konsole können Sie die Version der Verbindung in der Verbindungstabelle auf der Verbindungsseite, auf der Verbindungsdetailseite und in der Verbindungstabelle auf der Jobdetailseite sehen.

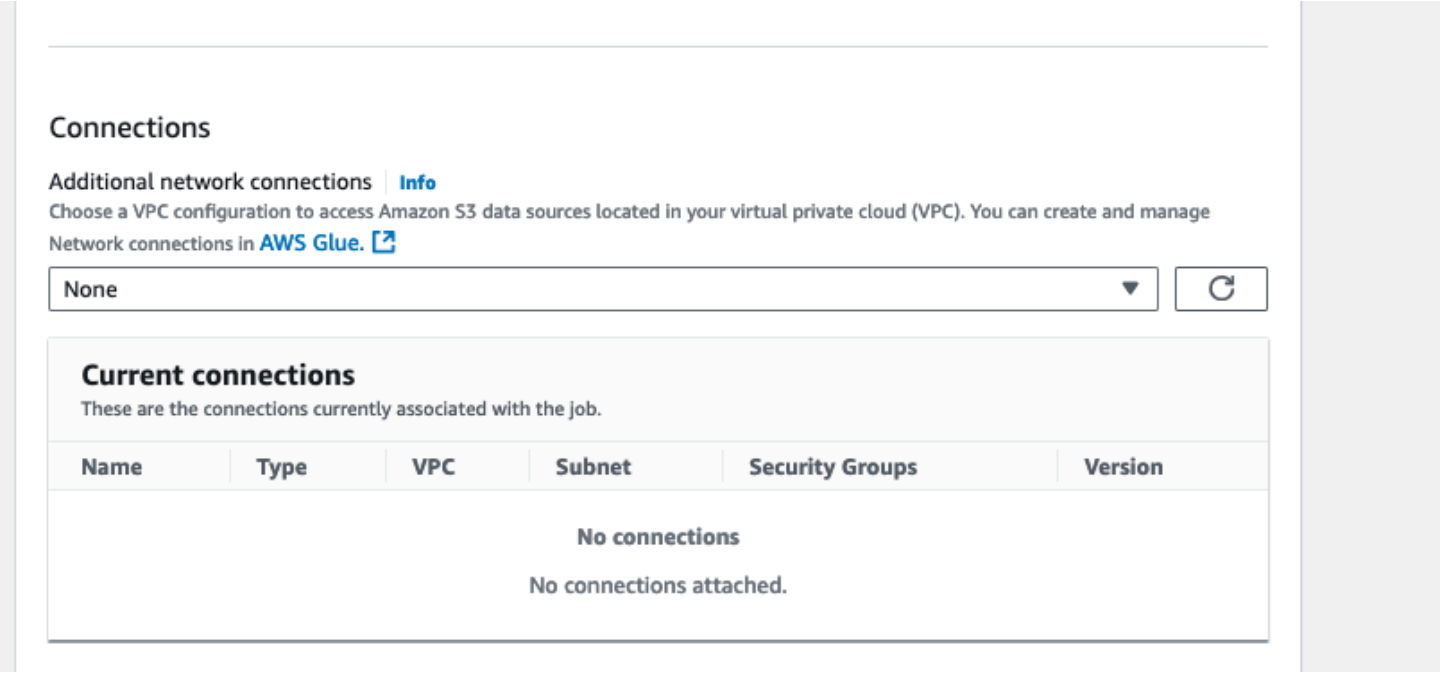
Die Verbindungsversion ist unter Verbindungsdetails sichtbar:



Die Verbindungsversion ist auch sichtbar, wenn Sie alle Ihre Verbindungen anzeigen.



Schließlich ist die Verbindungsversion auf der Registerkarte Jobdetails für einen Job sichtbar.



Bei Verbindungen der Version 2 stehen Ihnen die folgenden erweiterten Datenkonnektivitätsfunktionen zur Verfügung:

- Erkennung von Verbindungstypen: Support für das Erstellen von Verbindungen mithilfe standardisierter Vorlagen. AWS Glue erkennt automatisch die Verbindungstypen, auf die Sie zugreifen können, sowie die erforderlichen und optionalen Eingaben für einen bestimmten Verbindungstyp.
- Wiederverwendbarkeit: Verbindungsdefinitionen, die in allen AWS Datenverarbeitungsmodulen und Tools wie AWS Glue Amazon Athena, und wiederverwendet werden können. Amazon SageMaker AI Verbindungen enthalten jetzt AthenaProperties,, PythonProperties die es ermöglichen SparkProperties, zusätzlich zu den allgemeinen Eigenschaften, die in gespeichert sind, spezifische Verbindungseigenschaften für die Rechenumgebung oder den Dienst anzugeben. ConnectionProperties Athena erstellt jetzt Verbindungen, AWS Glue indem sie Athena-spezifische Eigenschaften in der AthenaProperties Eigenschaftenübersicht angibt.
- Datenvorschau: Möglichkeit, Metadaten zu durchsuchen und eine Vorschau von Daten aus verbundenen Quellen anzuzeigen.
- Konnektor-Metadaten: Wiederverwendbare Verbindungen können verwendet werden, um Tabellenmetadaten zu ermitteln.
- Mit Diensten verknüpfte Geheimnisse: Benutzer können in der `CreateConnection` Anfrage die erforderlichen OAuth, grundlegenden oder benutzerdefinierten Authentifizierungsdaten angeben. Die CreateConnection API erstellt ein Service Linked Secret in Ihrem Konto und speichert die Anmeldeinformationen in Ihrem Namen.

Unterstützte Authentifizierungstypen

Unified Connections unterstützt die folgenden Authentifizierungstypen:

- BASIC — Die meisten Datenbankverbindungstypen und vorhandenen AWS Glue Verbindungstypen unterstützen die Standardauthentifizierung, bei der es sich um einen Benutzernamen und ein Passwort handelt. Bisher war die Benennung der Schlüssel konnektorspezifisch und könnte beispielsweise user, username, userName, opensearch.net.http.auth.user usw. lauten. SecretsManager Hier standardisierte Unified Connections die grundlegenden Verbindungstypen für die Authentifizierung anhand der Schlüssel USERNAME und PASSWORD.
- OAUTH2— Die meisten neu eingeführten SaaS-Verbindungstypen unterstützen OAuth2 das Protokoll.

- BENUTZERDEFINIERT — Einige Verbindungstypen verfügen über einen anderen Authentifizierungsmechanismus wie Google, BigQuery bei dem von Benutzern erwartet wird, dass sie das von Google erhaltene JSON angeben BigQuery.

Überlegungen

Beachten Sie beim Erstellen einer einheitlichen Verbindung für Datenquellen die folgenden Unterschiede:

- Wenn Sie eine einheitliche Verbindung über herstellen AWS Glue Studio, werden Benutzeranmeldeinformationen AWS Secrets Manager nicht in der Verbindung selbst gespeichert. Das bedeutet, dass Jobs jetzt Zugriff auf Secrets Manager benötigen.
- Wenn Jobs in einer VPC ausgeführt werden, benötigen sie entweder einen VPC-Endpunkt oder ein NAT-Gateway für den Zugriff auf AWS Secrets Manager Secure Token Service (STS), was zusätzliche Kosten verursacht.
- Für bestimmte Datenquellen (Redshift, SQL Server, MySQL, Oracle, PostgreSQL) AWS Glue Studio erfordert die Erstellung einer einheitlichen Verbindung über Zugriff auf und. AWS STS AWS Secrets Manager Dies ist erforderlich, um eine sichere Verbindung herzustellen und die erforderlichen Anmeldeinformationen für den Zugriff auf diese Datenquellen in Ihrer Virtual Private Cloud (VPC) abzurufen.
- Für die Erstellung einer einheitlichen Verbindung ist eine IAM-Rolle mit Berechtigungen für den Zugriff auf AWS Secrets Manager und die Verwaltung von VPC-Ressourcen AWS Glue Studio erforderlich (falls Sie eine VPC verwenden):
  - Geheimnismanager: GetSecretValue
  - Geheimnismanager: PutSecretValue
  - Geheimnismanager: DescribeSecret
  - ec2: CreateNetworkInterface
  - ec2: DeleteNetworkInterface
  - ec2: DescribeNetworkInterfaces
  - ec2: DescribeSubnets

Verfügbare Verbindungen

AWS Glue unterstützt die folgenden Verbindungstypen:

- Adobe Analytics
- Adobe Marketo Engage
- Amazon Aurora (wird unterstützt, wenn der native JDBC-Treiber verwendet wird). (Nicht alle Treiberfunktionen können genutzt werden)
- Amazon DocumentDB
- Amazon-DynamoDB
- Amazon OpenSearch Service, zur Verwendung mit AWS Glue für Spark.
- Amazon Redshift
- Asana
- Azure Cosmos, für die Verwendung von Azure Cosmos DB für NoSQL mit ETL-Jobs AWS Glue
- Azure SQL, zur Verwendung mit AWS Glue für Spark.
- Blackbaud
- CircleCI
- Datadog
- Docusign-Monitor
- Domo
- Dynatrace
- Facebook Ads
- Facebook Page Insights
- Freshdesk
- Frische Verkäufe
- Google Ads
- Google Analytics 4
- Google BigQuery, zur Verwendung mit AWS Glue für Spark.
- Google-Suchkonsole
- Google Tabellen
- HubSpot
- Instagram Ads
- Intercom

- JDBC
- Jira Cloud
- Kafka
- Kunde
- LinkedIn
- Mailchimp
- Microsoft Dynamics 365 CRM
- Microsoft Teams
- Mixpanel
- Montag
- MongoDB
- MongoDB Atlas
- Okta
- Orakel NetSuite
- Paypal
- Pendo
- Pipedrive
- Produktboard
- QuickBooks
- Salesforce
- Salesforce Commerce Cloud
- Salesforce Marketing Cloud
- Kundenbindung in Salesforce Marketing Cloud (zuvor Salesforce Pardot)
- SAP HANA, zur Verwendung mit AWS Glue Spark.
- SAP OData
- SendGrid
- ServiceNow
- Slack

- Smartsheet
- Snapchat Ads
- Stripe
- Snowflake, zur Verwendung mit AWS Glue für Spark.
- Teradata Vantage, bei Verwendung für Spark. AWS Glue
- Twilio
- Vertica, zur Verwendung mit AWS Glue für Spark.
- WooCommerce
- Zendesk
- Zoho CRM
- Zoom Meetings
- Verschiedene Amazon Relational Database Service (Amazon RDS)-Angebote
- Network (Netzwerk) – Bezeichnet eine Netzwerkverbindung zu einer Datenquelle in einer Amazon Virtual Private Cloud (Amazon VPC).

Mit AWS Glue Studio können Sie auch eine Verbindung für einen Connector herstellen. Ein Konnektor ist ein optionales Codepaket für den Zugriff auf Datenspeicher in AWS Glue Studio. Weitere Informationen finden Sie unter Verwenden von Konnektoren und Verbindungen mit AWS Glue Studio

Informationen zum Herstellen einer Verbindung zu lokalen Datenbanken finden Sie auf der AWS Big Data Blog-Website unter So greifen Sie auf lokale Datenspeicher AWS Glue zu und analysieren diese mithilfe.

Um eine Verbindung mit VPC-Konfiguration herzustellen und gleichzeitig eine benutzerdefinierte IAM-Rolle zu verwenden, muss sie über die folgenden VPC-Zugriffsaktionen verfügen:

- Secretsmanager: GetSecretValue
- Geheimnismanager: PutSecretValue
- Geheimnismanager: DescribeSecret
- ec2: CreateNetworkInterface
- ec2: DeleteNetworkInterface
- ec2: DescribeNetworkInterfaces

- ec2: DescribeSubnets

Einschränkungen

- Sie können keine Verbindungen über die AWS Glue Konsole bearbeiten, wenn Sie eine v2-Verbindung erstellt haben mit AWS Glue APIs:
  - Amazon DocumentDB
  - Amazon Aurora
  - MariaDB
  - MongoDB Atlas
  - MongoDB

AWS Glue Verbindungseigenschaften

Dieses Thema enthält Informationen zu Eigenschaften für AWS Glue Verbindungen.

Themen

- Erforderliche Verbindungseigenschaften
- AWS Glue Eigenschaften der JDBC-Verbindung
- AWS Glue Verbindungseigenschaften von MongoDB und MongoDB Atlas
- Eigenschaften der Salesforce-Verbindung
- Snowflake-Verbindung
- Vertica-Verbindung
- SAP-HANA-Verbindung
- Azure-SQL-Verbindung
- Teradata-Vantage-Verbindung
- OpenSearch Verbindung zum Dienst
- Azure-Cosmos-Verbindung
- AWS Glue Eigenschaften der SSL-Verbindung
- Apache-Kafka-Verbindungseigenschaften für die Client-Authentifizierung
- BigQuery Google-Verbindung

- Vertica-Verbindung

Erforderliche Verbindungseigenschaften

Wenn Sie eine Verbindung auf der AWS Glue Konsole definieren, müssen Sie Werte für die folgenden Eigenschaften angeben:

Verbindungsname

Geben Sie einen eindeutigen Namen für Ihre Verbindung ein.

Verbindungstyp

Wählen Sie JDBC oder einen der spezifischen Verbindungstypen.

Weitere Informationen zum JDBC-Verbindungstyp finden Sie unter the section called “Eigenschaften der JDBC-Verbindung”

Wählen Sie Network (Netzwerk) aus, um eine Datenquelle in einer Amazon Virtual Private Cloud-Umgebung (Amazon VPC) zu verbinden.

Je nach ausgewähltem Typ zeigt die AWS Glue -Konsole weitere erforderliche Felder an. Wenn Sie beispielsweise Amazon RDS auswählen, müssen Sie anschließend die Datenbank-Engine auswählen.

SSL-Verbindung erforderlich

Wenn Sie diese Option auswählen, AWS Glue müssen Sie sicherstellen, dass die Verbindung zum Datenspeicher über ein vertrauenswürdiges Secure Sockets Layer (SSL) hergestellt wird.

Weitere Informationen, einschließlich zusätzlicher Optionen, die bei Auswahl dieser Option verfügbar sind, finden Sie unter the section called “SSL-Verbindungseigenschaften”.

Wählen Sie MSK-Cluster (Amazon Managed Streaming for Apache Kafka)

Gibt einen MSK-Cluster von einem anderen AWS Konto an.

Kafka-Bootstrap-Server URLs (nur Kafka)

Gibt eine durch Kommas getrennte Liste von Bootstrap-Servern an. URLs Schließen Sie die Portnummer ein. Beispiel: b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-2.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-3.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094

AWS Glue Eigenschaften der JDBC-Verbindung

AWS Glue Studio erstellt jetzt einheitliche Verbindungen für MySQL-, Oracle-, PostgresSQL-, Redshift- und SQL Server-Datenquellen, was zusätzliche Schritte für den Zugriff auf Secrets Manager- und VPC-Ressourcen erfordert, was zusätzliche Kosten verursachen kann. Sie können auf diese Verbindungen zugreifen, AWS Glue Studio indem Sie den Verbindungsnamen für die jeweilige Verbindung auswählen.

Weitere Informationen finden Sie unter Überlegungen.

AWS Glue kann über eine JDBC-Verbindung eine Verbindung zu den folgenden Datenspeichern herstellen:

- Amazon Redshift
- Amazon Aurora
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL
- Snowflake, wenn Crawler verwendet werden. AWS Glue
- Aurora (unterstützt, wenn der native JDBC-Treiber verwendet wird. Nicht alle Features des Treibers können genutzt werden)
- Amazon RDS for MariaDB

> **Important**
>
> Ein ETL-Auftrag kann derzeit nur eine JDBC-Verbindung mit einem Subnetz verwenden. Wenn Sie über mehrere Datenspeicher in einem Auftrag verfügen, müssen sich diese im gleichen Subnetz befinden oder vom Subnetz erreichbar sein.
> Wenn Sie sich dafür entscheiden, Ihre eigenen JDBC-Treiberversionen für AWS Glue Crawler zu verwenden, verbrauchen Ihre Crawler Ressourcen in AWS Glue Jobs und Amazon S3, um sicherzustellen, dass Ihre bereitgestellten Treiber in Ihrer Umgebung ausgeführt werden. Der zusätzliche Ressourcenverbrauch wird in Ihrem Konto angezeigt. Darüber hinaus bedeutet die Bereitstellung eines eigenen JDBC-Treibers nicht, dass

der Crawler alle Features des Treibers nutzen kann. Treiber sind auf die Eigenschaften beschränkt, die unter Definieren von Verbindungen im Data Catalog beschrieben sind.

Im Folgenden finden Sie zusätzliche Eigenschaften für den JDBC-Verbindungstyp.

JDBC-URL

Geben Sie die URL für Ihren JDBC-Datenspeicher ein. Für die meisten Datenbank-Engines wird dieses Feld in folgendem Format angegeben. Ersetzen Sie in diesem Format, *protocol* *hostport*, und *db_name* durch Ihre eigenen Informationen.

```
jdbc:protocol://host:port/db_name
```

Abhängig von der Datenbank-Engine kann jedoch ein anderes JDBC-URL-Format erforderlich sein. Dieses Format kann im Hinblick auf die Nutzung des Doppelpunkts (:) und Schrägstrichs (/) oder die Schlüsselwörter, mit denen Datenbanken angegeben werden, geringfügig abweichen.

Damit JDBC eine Verbindung mit dem Datenspeicher herstellen kann, ist ein `db_name` im Datenspeicher erforderlich. Der `db_name` wird verwendet, um eine Netzwerkverbindung mit dem bereitgestellten `username` und `password` herzustellen. Wenn eine Verbindung besteht, AWS Glue kann auf andere Datenbanken im Datenspeicher zugegriffen werden, um einen Crawler oder einen ETL-Job auszuführen.

Die folgenden JDBC-URL-Beispiele veranschaulichen die Syntax für mehrere Datenbank-Engines.

- Herstellen einer Verbindung mit einem Amazon-Redshift-Cluster-Datenspeicher mithilfe einer `dev`-Datenbank:

  ```
  jdbc:redshift://xxx.us-east-1.redshift.amazonaws.com:8192/dev
  ```

- Herstellen einer Verbindung mit einem Amazon RDS for MySQL-Datenspeicher mithilfe einer `employee`-Datenbank:

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:3306/
  employee
  ```

- Herstellen einer Verbindung mit einem Amazon RDS for PostgreSQL-Datenspeicher mithilfe einer `employee`-Datenbank:

```
jdbc:postgresql://xxx-cluster.cluster-xxx.us-
east-1.rds.amazonaws.com:5432/employee
```

- Herstellen einer Verbindung mit einem Amazon RDS for Oracle-Datenspeicher mithilfe eines `employee`-Servicenamens:

  ```
  jdbc:oracle:thin://@xxx-cluster.cluster-xxx.us-
  east-1.rds.amazonaws.com:1521/employee
  ```

  Die Syntax für Amazon RDS for Oracle kann das folgende Muster aufweisen: Ersetzen Sie in diesen Mustern*host*, *portservice_name*, und *SID* durch Ihre eigenen Informationen.

  - `jdbc:oracle:thin://@`*host*`:`*port*`/`*service_name*
  - `jdbc:oracle:thin://@`*host*`:`*port*`:`*SID*

- Eine Verbindung mit einem Amazon RDS for Microsoft SQL Server-Datenspeicher mit einer `employee`-Datenbank herstellen:

  ```
  jdbc:sqlserver://xxx-cluster.cluster-xxx.us-
  east-1.rds.amazonaws.com:1433;databaseName=employee
  ```

  Die Syntax für Amazon RDS for SQL Server kann das folgende Muster aufweisen: Ersetzen Sie in diesen Mustern *server_nameport*, und *db_name* durch Ihre eigenen Informationen.

  - `jdbc:sqlserver://`*server_name*`:`*port*`;database=`*db_name*
  - `jdbc:sqlserver://`*server_name*`:`*port*`;databaseName=`*db_name*

- Um eine Verbindung mit einer Amazon Aurora PostgreSQL `employee` Datenbankinstanz herzustellen, geben Sie den Endpunkt für die Datenbankinstanz, den Port und den Datenbanknamen an:

  ```
  jdbc:postgresql://employee_instance_1.xxxxxxxxxxxx.us-
  east-2.rds.amazonaws.com:5432/employee
  ```

- Um eine Verbindung zu einem Amazon RDS for MariaDB Datenspeicher mit einer `employee` Datenbank herzustellen, geben Sie den Endpunkt für die Datenbankinstanz, den Port und den Datenbanknamen an:

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.aws-
  region.rds.amazonaws.com:3306/employee
  ```

- > **Warning**
  >
  > Snowflake-JDBC-Verbindungen werden nur von Crawlern unterstützt. AWS Glue Wenn Sie den Snowflake-Connector in AWS Glue Jobs verwenden, verwenden Sie den Snowflake-Verbindungstyp.

  Geben Sie zum Herstellen einer Verbindung mit einer Snowflake-Instance der `sample`-Datenbank den Endpunkt für die Snowflake-Instance, den Benutzer, den Datenbanknamen und den Rollennamen an. Sie können optional den `warehouse`-Parameter hinzufügen.

  ```
  jdbc:snowflake://account_name.snowflakecomputing.com/?
  user=user_name&db=sample&role=role_name&warehouse=warehouse_name
  ```

  > **Important**
  >
  > Bei Snowflake-Verbindungen über JDBC wird die Reihenfolge der Parameter in der URL erzwungen und muss wie folgt geordnet werden: `user`, db, `role_name` und `warehouse`.

- Um eine Verbindung zu einer Snowflake-Instanz der `sample` Datenbank mit einem AWS privaten Link herzustellen, geben Sie die Snowflake-JDBC-URL wie folgt an:

  ```
  jdbc:snowflake://account_name.region.privatelink.snowflakecomputing.com/?
  user=user_name&db=sample&role=role_name&warehouse=warehouse_name
  ```

Username

> **Note**
>
> Wir empfehlen, dass Sie zum Speichern der Verbindungsdaten ein AWS Geheimnis verwenden, anstatt Ihren Benutzernamen und Ihr Passwort direkt anzugeben. Weitere Informationen finden Sie unter Speichern von Verbindungsanmeldedaten in AWS Secrets Manager.

Geben Sie einen Benutzernamen ein, der über die Berechtigung für den Zugriff auf den JDBC-Datenspeicher verfügt.

Passwort

Geben Sie das Passwort für den Benutzernamen ein, der über Zugriffsberechtigungen für den JDBC-Datenspeicher verfügt.

Port

Geben Sie den Port ein, der in der JDBC-URL verwendet wird, um eine Verbindung mit einer Amazon-RDS-Oracle-Instance herzustellen. Dieses Feld wird nur angezeigt, wenn Require SSL connection (SSL-Verbindung anfordern) für eine Amazon-RDS-Oracle-Instance ausgewählt ist.

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

> **Important**
>
> Wenn Sie über eine JDBC-Verbindung arbeiten AWS, von der aus gehostet wird, z. B. mit Daten von Snowflake, sollte Ihre VPC über ein NAT-Gateway verfügen, das den Datenverkehr in öffentliche und private Subnetze aufteilt. Das öffentliche Subnetz wird für die Verbindung mit der externen Quelle verwendet, und das interne Subnetz wird für die Verarbeitung von verwendet. AWS Glue Informationen zum Konfigurieren Ihrer Amazon VPC für externe Verbindungen finden Sie unter Herstellen einer Verbindung mit dem Internet oder anderen Netzwerken mithilfe von NAT-Geräten und Einrichtung von Amazon VPC für JDBC-Verbindungen zu Amazon RDS-Datenspeichern von AWS Glue.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

Name der JDBC-Treiberklasse – optional

Geben Sie den Namen der benutzerdefinierten JDBC-Treiberklasse an:

- Postgres – org.postgresql.Driver
- MySQL – com.mysql.jdbc.Driver, com.mysql.cj.jdbc.Driver
- Redshift – com.amazon.redshift.jdbc.Driver, com.amazon.redshift.jdbc42.Driver
- Oracle — oracle.jdbc.driver. OracleDriver
- SQL Server — com.microsoft.sqlserver.jdbc. SQLServerTreiber

JDBC-Treiber-S3-Pfad – optional

Geben Sie den Amazon-S3-Speicherort für den benutzerdefinierten JDBC-Treiber an. Dies ist ein absoluter Pfad zu einer .jar-Datei. Wenn Sie Ihre eigenen JDBC-Treiber bereitstellen möchten, um eine Verbindung zu Ihren Datenquellen für Ihre vom Crawler unterstützten Datenbanken herzustellen, können Sie Werte für Parameter `customJdbcDriverS3Path` und `customJdbcDriverClassName` angeben.
Die Verwendung eines vom Kunden bereitgestellten JDBC-Treibers ist auf das erforderliche Erforderliche Verbindungseigenschaften beschränkt.

AWS Glue Verbindungseigenschaften von MongoDB und MongoDB Atlas

Im Folgenden finden Sie zusätzliche Eigenschaften für den MongoDB- oder MongoDB-Atlas-Verbindungstyp.

MongoDB-URL

Geben Sie die URL für Ihren MongoDB- oder MongoDB-Atlas-Datenspeicher ein:

- Für MongoDB: mongodb://host:port/database. Der Host kann ein Hostname, eine IP-Adresse oder ein UNIX-Domain-Socket sein. Wenn die Verbindungszeichenfolge keinen Port angibt, wird der standardmäßige MongoDB-Port 27017 verwendet.
- Für MongoDB Atlas: mongodb+srv://server.example.com/database. Der Host kann ein Hostname sein, der im Folgenden einem DNS-SRV-Eintrag entspricht. Das SRV-Format benötigt keinen Port und verwendet den standardmäßigen MongoDB-Port 27017.

Username

> **Note**
>
> Wir empfehlen, dass Sie zum Speichern der Verbindungsanmeldeinformationen einen AWS geheimen Schlüssel verwenden, anstatt Ihren Benutzernamen und Ihr Passwort direkt einzugeben. Weitere Informationen finden Sie unter Speichern von Verbindungsanmeldedaten in AWS Secrets Manager.

Geben Sie einen Benutzernamen ein, der über die Berechtigung für den Zugriff auf den JDBC-Datenspeicher verfügt.

Passwort

Geben Sie das Passwort für den Benutzernamen ein, der über Zugriffsberechtigungen für den MongoDB- oder MongoDB-Atlas-Datenspeicher verfügt.

Eigenschaften der Salesforce-Verbindung

Im Folgenden finden Sie zusätzliche Eigenschaften für den Salesforce-Verbindungstyp.

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Salesforce.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Salesforce Rest API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `IMPORT_DELETED_RECORDS`(Zeichenfolge) — Standard: FALSE. Wird zum Lesen verwendet. Um die gelöschten Datensätze während der Abfrage abzurufen.
- `WRITE_OPERATION`(Zeichenfolge) — Standard: INSERT. Wird zum Schreiben verwendet. Der Wert sollte INSERT, UPDATE, UPSERT, DELETE sein.
- `ID_FIELD_NAMES`(Zeichenfolge) — Standard: null. Wird nur für UPSERT verwendet.

Snowflake-Verbindung

Die folgenden Eigenschaften werden verwendet, um eine Snowflake-Verbindung einzurichten, die in AWS Glue ETL-Jobs verwendet wird. Verwenden Sie beim Crawling von Snowflake eine JDBC-Verbindung.

Snowflake-URL

Die URL Ihres Snowflake-Endpunktes. Weitere Informationen zum Snowflake-Endpunkt URLs finden Sie unter Connecting to Your Accounts in der Snowflake-Dokumentation.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mit den `sfPassword` Tasten `sfUser` und und Ihres Geheimnisses eine Verbindung zu Snowflake her.

Snowflake-Rolle (optional)

Beim Herstellen der Verbindung AWS Glue wird eine Snowflake-Sicherheitsrolle verwendet.

Verwenden Sie die folgenden Eigenschaften, wenn Sie eine Verbindung zu einem Snowflake-Endpunkt konfigurieren, der in Amazon VPC mithilfe von AWS PrivateLink gehostet wird.

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

Vertica-Verbindung

Verwenden Sie die folgenden Eigenschaften, um eine Vertica-Verbindung für AWS Glue ETL-Jobs einzurichten.

Vertica-Host

Der Hostname Ihrer Vertica-Installation.

Vertica-Port

Der Port, über den Ihre Vertica-Installation verfügbar ist.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mit den Schlüsseln Ihres Geheimnisses eine Verbindung zu Vertica her.

Verwenden Sie die folgenden Eigenschaften, wenn Sie eine Verbindung zu einem Vertica-Endpunkt konfigurieren, der in Amazon VPC gehostet wird.

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

SAP-HANA-Verbindung

Verwenden Sie die folgenden Eigenschaften, um eine SAP HANA-Verbindung für AWS Glue ETL-Jobs einzurichten.

SAP-HANA-URL

EINE SAP-JDBC-URL.

SAP HANA JDBC URLs haben das folgende Format `jdbc:sap://saphanaHostname:saphanaPort/?databaseName=saphanaDBname,ParameterNa`

AWS Glue erfordert die folgenden JDBC-URL-Parameter:

- `databaseName` – Eine Standarddatenbank in SAP HANA, mit der eine Verbindung hergestellt werden kann.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mithilfe der Schlüssel Ihres Geheimnisses eine Verbindung zu SAP HANA her.

Verwenden Sie die folgenden Eigenschaften, wenn Sie eine Verbindung zu einem SAP-HANA-Endpunkt konfigurieren, der in Amazon VPC gehostet wird:

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

Azure-SQL-Verbindung

Verwenden Sie die folgenden Eigenschaften, um eine Azure SQL-Verbindung für AWS Glue ETL-Jobs einzurichten.

Azure-SQL-URL

Die JDBC-URL eines Azure-SQL-Endpunkts.

Die URL muss das folgende Format aufweisen: `jdbc:sqlserver://databaseServerName:databasePort;databaseName=azuresqlDBname;`.

AWS Glue erfordert die folgenden URL-Eigenschaften:

- `databaseName` – Eine Standarddatenbank in Azure SQL, mit der eine Verbindung hergestellt werden kann.

Weitere Informationen zu JDBC URLs für Azure SQL Managed Instances finden Sie in der Microsoft-Dokumentation.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mithilfe der Schlüssel Ihres Geheimnisses eine Verbindung zu Azure SQL her.

Teradata-Vantage-Verbindung

Verwenden Sie die folgenden Eigenschaften, um eine Teradata Vantage-Verbindung für AWS Glue ETL-Jobs einzurichten.

Teradata-URL

Um eine Verbindung zu einer Teradata-Instance herzustellen, geben Sie den Hostnamen der Datenbank-Instance und die entsprechenden Teradata-Parameter an:

```
jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName=Pa
```

AWS Glue unterstützt die folgenden JDBC-URL-Parameter:

- `DATABASE_NAME` – Eine Standarddatenbank in Teradata, mit der eine Verbindung hergestellt werden kann.
- `DBS_PORT` – Gibt den Teradata-Port an, falls dieser vom Standardwert abweicht.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mithilfe der Schlüssel Ihres Geheimnisses eine Verbindung zu Teradata Vantage her.

Verwenden Sie die folgenden Eigenschaften, wenn Sie eine Verbindung zu einem Teradata-Vantage-Endpunkt konfigurieren, der in Amazon VPC gehostet wird:

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet

diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

OpenSearch Verbindung zum Dienst

Verwenden Sie die folgenden Eigenschaften, um eine OpenSearch Dienstverbindung für AWS Glue ETL-Jobs einzurichten.

Domains-Endpunkte

Ein Amazon OpenSearch Service-Domain-Endpunkt hat das folgende Standardformat: https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com. Weitere Informationen zur Identifizierung Ihres Domain-Endpunkts finden Sie unter Amazon OpenSearch Service-Domains erstellen und verwalten in der Amazon OpenSearch Service-Dokumentation.

Port

Der offene Port im Endpunkt.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mit den Schlüsseln Ihres Geheimnisses eine Verbindung zum OpenSearch Service her.

Verwenden Sie die folgenden Eigenschaften, wenn Sie eine Verbindung zu einem in Amazon VPC gehosteten OpenSearch Service-Endpunkt konfigurieren:

VPC

Wählen Sie den Namen der Virtual Private Cloud (VPC) aus, die den Datenspeicher enthält. Die AWS Glue Konsole listet alle VPCs für die aktuelle Region auf.

Subnetz

Wählen Sie das Subnetz in der VPC aus, die den Datenspeicher enthält. Die AWS Glue -Konsole listet alle Subnetze für den Datenspeicher in Ihrer VPC auf.

Sicherheitsgruppen

Wählen Sie die Sicherheitsgruppen aus, die Ihrem Datenspeicher zugeordnet sind. AWS Glue erfordert eine oder mehrere Sicherheitsgruppen mit einer Regel für eingehende Quellen, die das Herstellen einer Verbindung AWS Glue ermöglicht. Die AWS Glue Konsole listet alle Sicherheitsgruppen auf, denen eingehender Zugriff auf Ihre VPC gewährt wird. AWS Glue ordnet

diese Sicherheitsgruppen der elastic network interface zu, die mit Ihrem VPC-Subnetz verbunden ist.

Azure-Cosmos-Verbindung

Verwenden Sie die folgenden Eigenschaften, um eine Azure Cosmos-Verbindung für AWS Glue ETL-Jobs einzurichten.

Endpunkt-URI für das Azure-Cosmos-DB-Konto

Der für die Verbindung mit Azure Cosmos verwendete Endpunkt. Weitere Informationen finden Sie in der Azure-Dokumentation.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue stellt mithilfe der Schlüssel Ihres Geheimnisses eine Verbindung zu Azure Cosmos her.

AWS Glue Eigenschaften der SSL-Verbindung

Im Folgenden finden Sie Details zur Eigenschaft SSL-Verbindung erforderlich.

Wenn Sie keine SSL-Verbindung benötigen, werden Fehler AWS Glue ignoriert, wenn SSL zum Verschlüsseln einer Verbindung zum Datenspeicher verwendet wird. In der Dokumentation Ihrer Datenbank finden Sie Konfigurationsanweisungen. Wenn Sie diese Option auswählen, schlagen die Auftragsausführungs-, Crawler- oder ETL-Anweisungen in einem Entwicklungsendpunkt fehl, wenn sich AWS Glue nicht verbinden kann.

> **Note**
>
> Snowflake unterstützt standardmäßig eine SSL-Verbindung, daher gilt diese Eigenschaft nicht für Snowflake.

Diese Option wird auf der AWS Glue Clientseite validiert. Bei JDBC-Verbindungen werden AWS Glue nur Verbindungen über SSL mit Zertifikat- und Hostnamenvalidierung hergestellt. SSL-Verbindungsunterstützung ist verfügbar für:

- Oracle Database

- Microsoft SQL Server
- PostgreSQL
- Amazon Redshift
- MySQL (nur für Amazon-RDS-Instances)
- Amazon Aurora MySQL (nur für Amazon-RDS-Instances)
- Amazon Aurora PostgreSQL (Nur Amazon RDS-Instances)
- Kafka, das beinhaltet Amazon Managed Streaming for Apache Kafka
- MongoDB

> **Note**
>
> Um einem Amazon-RDS-Oracle--Datenspeicher die Verwendung von Require SSL connection (SSL-Verbindung anfordern) zu ermöglichen, müssen Sie eine Optionsgruppe erstellen und an die Oracle-Instance anfügen.
>
> 1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon RDS-Konsole unter https://console.aws.amazon.com/rds/.
> 2. Fügen Sie eine Optionsgruppe zur Amazon-RDS-Oracle-Instance hinzu. Weitere Informationen zum Hinzufügen einer Optionsgruppe auf der Amazon-RDS-Konsole finden Sie unter Creating an Option Group (Erstellen einer Optionsgruppe).
> 3. Hinzufügen einer Option zur Optionsgruppe für SSL. Der Port, den Sie für SSL angeben, wird später verwendet, wenn Sie eine AWS Glue JDBC-Verbindungs-URL für die Amazon RDS-Oracle-Instance erstellen. Weitere Informationen zum Hinzufügen einer Option auf der Amazon-RDS-Konsole finden Sie unter Adding an Option to an Option Group (Hinzufügen einer Option zu einer Optionsgruppe) im Benutzerhandbuch für Amazon RDS. Weitere Informationen zu den Oracle-SSL-Optionen finden Sie unter Oracel SSL im Amazon RDS-Benutzerhandbuch.
> 4. Stellen Sie auf der AWS Glue Konsole eine Verbindung zur Amazon RDS Oracle-Instance her. Wählen Sie in der Verbindungsdefinition die Option SSL-Verbindung erforderlichaus. Geben Sie bei Bedarf den Port ein, den Sie in der Option Amazon RDS Oracle SSL verwendet haben.

Die folgenden zusätzlichen optionalen Eigenschaften sind verfügbar, wenn Require SSL connection (SSL-Verbindung erforderlich) für eine Verbindung ausgewählt ist.

Benutzerdefiniertes JDBC-Zertifikat in S3

Wenn Sie über ein Zertifikat verfügen, das Sie derzeit für die SSL-Kommunikation mit Ihren lokalen oder Cloud-Datenbanken verwenden, können Sie dieses Zertifikat für SSL-Verbindungen zu AWS Glue Datenquellen oder Zielen verwenden. Geben Sie einen Amazon Simple Storage Service (Amazon S3) -Standort ein, der ein benutzerdefiniertes Stammzertifikat enthält. AWS Glue verwendet dieses Zertifikat, um eine SSL-Verbindung zur Datenbank herzustellen. AWS Glue verarbeitet nur X.509-Zertifikate. Das Zertifikat muss DER-codiert sein und im base64-PEM-Codierungsformat bereitgestellt werden.

Wenn dieses Feld leer bleibt, wird das Standardzertifikat verwendet.

Benutzerdefinierte JDBC-Zertifikatzeichenfolge

Geben Sie JDBC-Datenbank-Zertifikatinformationen ein. Diese Zeichenfolge wird für den Domainabgleich oder den Distinguished Name (DN)-Abgleich verwendet wird. Im Fall von Oracle Database entspricht die Zeichenfolge dem Parameter `SSL_SERVER_CERT_DN` im Sicherheitsabschnitt der Datei `tnsnames.ora`. Im Fall von Microsoft SQL Server wird diese Zeichenfolge als `hostNameInCertificate` verwendet.

Im Folgenden finden Sie ein Beispiel für den Oracle Database-Parameter `SSL_SERVER_CERT_DN`.

```
cn=sales,cn=OracleContext,dc=us,dc=example,dc=com
```

Speicherort des privaten CA-Zertifikats von Kafka

Wenn Sie über ein Zertifikat verfügen, das Sie derzeit für die SSL-Kommunikation mit Ihrem Kafka-Datenspeicher verwenden, können Sie dieses Zertifikat mit Ihrer AWS Glue Verbindung verwenden. Diese Option ist für Kafka-Datenspeicher erforderlich und für Amazon Managed Streaming for Apache Kafka Datenspeicher optional. Geben Sie einen Amazon Simple Storage Service (Amazon S3) -Standort ein, der ein benutzerdefiniertes Stammzertifikat enthält. AWS Glue verwendet dieses Zertifikat, um eine SSL-Verbindung zum Kafka-Datenspeicher herzustellen. AWS Glue verarbeitet nur X.509-Zertifikate. Das Zertifikat muss DER-codiert sein und im base64-PEM-Codierungsformat bereitgestellt werden.

Zertifikatvalidierung überspringen

Aktivieren Sie das Kontrollkästchen Zertifikatsvalidierung überspringen, um die Validierung des benutzerdefinierten Zertifikats zu AWS Glueüberspringen. Wenn Sie sich für die Validierung entscheiden, werden der Signaturalgorithmus und der Betreff-Public-Key-Algorithmus für

das Zertifikat AWS Glue validiert. Wenn das Zertifikat nicht validiert werden kann, schlägt die Ausführung aller ETL-Aufgaben oder Crawler fehl, die diese Verbindung verwenden.

Die einzig zulässigen Signaturalgorithmen sind SHA256with RSA, RSA SHA384with oder RSA. SHA512with Die Schlüssellänge für den Algorithmus für den öffentlichen Schlüssel muss mindestens 2048 betragen.

Kafka-Client-Keystore-Speicherort

Der Amazon-S3-Speicherort der Client-Keystore-Datei für die clientseitige Kafka-Authentifizierung. Der Pfad muss das Format s3://.jks haben. bucket/prefix/filename Er muss mit dem Dateinamen und der Erweiterung .jks enden.

Passwort für den Kafka-Client-Keystore (optional)

Das Passwort für den Zugriff auf den bereitgestellten Keystore.

Passwort für den Kafka-Client-Schlüssel (optional)

Ein Keystore kann aus mehreren Schlüsseln bestehen, also ist dies das Passwort für den Zugriff auf den Clientschlüssel, der mit dem serverseitigen Kafka-Schlüssel verwendet werden soll.

Apache-Kafka-Verbindungseigenschaften für die Client-Authentifizierung

AWS Glue unterstützt das SASL-Framework (Simple Authentication and Security Layer) für die Authentifizierung, wenn Sie eine Apache Kafka-Verbindung herstellen. Das SASL-Framework unterstützt verschiedene Authentifizierungsmechanismen und AWS Glue bietet die Protokolle SCRAM (Benutzername und Passwort), GSSAPI (Kerberos-Protokoll) und PLAIN.

Wird verwendet AWS Glue Studio , um eine der folgenden Client-Authentifizierungsmethoden zu konfigurieren. Weitere Informationen finden Sie im AWS Glue Studio Benutzerhandbuch unter Verbindungen für Konnektoren erstellen.

- Keine – Keine Authentifizierung. Dies ist nützlich, wenn Sie eine Verbindung zu Testzwecken herstellen.
- SASL/SCRAM-SHA-512 – Wenn Sie diese Authentifizierungsmethode wählen, können Sie Anmeldeinformationen zur Authentifizierung angeben. Es gibt zwei Optionen:
  - AWS Secrets Manager verwenden (empfohlen) — wenn Sie diese Option wählen, können Sie Ihren Benutzernamen und Ihr Passwort in AWS Secrets Manager speichern und bei Bedarf AWS Glue darauf zugreifen lassen. Geben Sie das Secret an, das die SSL- oder SASL-

Authentifizierungsdaten speichert. Weitere Informationen finden Sie unter Speichern von Verbindungsanmeldedaten in AWS Secrets Manager.

- Geben Sie einen Benutzernamen und ein Passwort ein.

- SASL/GSSAPI (Kerberos) - if you select this option, you can select the location of the keytab file, krb5.conf file and enter the Kerberos principal name and Kerberos service name. The locations for the keytab file and krb5.conf file must be in an Amazon S3 location. Since MSK does not yet support SASL/GSSAPI, diese Option ist nur für vom Kunden verwaltete Apache Kafka-Cluster verfügbar. Weitere Informationen finden Sie unter MIT Kerberos-Dokumentation: Keytab.
- SASL/PLAIN — Wählen Sie diese Authentifizierungsmethode, um die Authentifizierungsdaten anzugeben. Es gibt zwei Optionen:
  - AWS Secrets Manager verwenden (empfohlen) — Wenn Sie diese Option wählen, können Sie Ihre Anmeldeinformationen in AWS Secrets Manager speichern und bei Bedarf AWS Glue auf die Informationen zugreifen lassen. Geben Sie das Secret an, das die SSL- oder SASL-Authentifizierungsdaten speichert.
  - Geben Sie den Benutzernamen und das Passwort direkt ein.
- SSL-Clientauthentifizierung – Wenn Sie diese Option auswählen, können Sie den Standort des Kafka-Client-Keystores auswählen, indem Sie Amazon S3 durchsuchen. Optional können Sie das Kennwort für den Kafka-Client-Keystore und das Kafka-Client-Schlüsselkennwort eingeben.

BigQuery Google-Verbindung

Die folgenden Eigenschaften werden verwendet, um eine BigQuery Google-Verbindung einzurichten, die in AWS Glue ETL-Jobs verwendet wird. Weitere Informationen finden Sie unter the section called “BigQuery Verbindungen”.

AWS Geheim

Der geheime Name eines Geheimnisses in AWS Secrets Manager. AWS Glue ETL-Jobs stellen BigQuery mithilfe des `credentials` Schlüssels Ihres Geheimnisses eine Verbindung zu Google her.

Vertica-Verbindung

Die folgenden Eigenschaften werden verwendet, um eine Vertica-Verbindung einzurichten, die in AWS Glue ETL-Jobs verwendet wird. Weitere Informationen finden Sie unter the section called “Vertica-Verbindungen”.

Speichern von Verbindungsanmeldedaten in AWS Secrets Manager

Wir empfehlen, dass Sie es verwenden, AWS Secrets Manager um Verbindungsanmeldedaten für Ihren Datenspeicher anzugeben. Wenn Sie Secrets Manager auf diese Weise verwenden, können Sie zur Laufzeit bei ETL-Jobs und Crawler-Läufen auf Ihr Geheimnis AWS Glue zugreifen und Ihre Anmeldeinformationen schützen.

Voraussetzungen

Um Secrets Manager mit verwenden zu können AWS Glue, müssen Sie Ihrer IAM-Rolle die AWS Glue Erlaubnis zum Abrufen geheimer Werte erteilen. Die AWS verwaltete Richtlinie beinhaltet `AWSGlueServiceRole` keine AWS Secrets Manager Berechtigungen. Beispiele für IAM-Richtlinien finden Sie unter Beispiel: Berechtigung zum Abrufen von Secret-Werten im AWS Secrets Manager-Benutzerhandbuch.

Abhängig von Ihrer Netzwerkkonfiguration müssen Sie möglicherweise auch einen VPC-Endpunkt erstellen, um eine private Verbindung zwischen Ihrer VPC und dem Secrets Manager herzustellen. Weitere Informationen finden Sie unter Verwenden von AWS Secrets Manager -VPC-Endpunkten.

Um ein Geheimnis zu erstellen für AWS Glue

1. Befolgen Sie die Anweisungen unter Create and manage secrets (Secrets erstellen und verwalten) im AWS Secrets Manager -Benutzerhandbuch. Das folgende Beispiel-JSON zeigt, wie Sie Ihre Anmeldeinformationen im Reiter Plaintext (Klartext) angeben, wenn Sie ein Secret für AWS Glue erstellen.

   ```
   {
     "username": "EXAMPLE-USERNAME",
     "password": "EXAMPLE-PASSWORD"
   }
   ```

2. Ordnen Sie Ihr Geheimnis über die AWS Glue Studio Schnittstelle einer Verbindung zu. Detaillierte Anweisungen finden Sie unter Erstellen von Verbindungen für Konnektoren im AWS Glue Studio -Benutzerhandbuch.

AWS Glue Verbindung hinzufügen

Sie können in AWS Glue Spark programmgesteuert eine Verbindung zu Datenquellen herstellen. Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

Sie können die AWS Glue Konsole auch verwenden, um Verbindungen hinzuzufügen, zu bearbeiten, zu löschen und zu testen. Informationen zu AWS Glue Verbindungen finden Sie unterHerstellen einer Verbindung zu Daten.

Themen

- Verbindung zu Adobe Analytics herstellen
- Verbindung zu Adobe Marketo Engage herstellen
- Verbindung zu Amazon Redshift herstellen in AWS Glue Studio
- Verbindung zu Asana herstellen
- Verbindung zu Azure Cosmos DB herstellen in AWS Glue Studio
- Verbindung zu Azure SQL herstellen in AWS Glue Studio
- Verbindung zu Blackbaud Raiser's Edge NXT herstellen
- Verbindung zu CircleCI herstellen
- Verbindung zu Datadog herstellen
- Verbindung zu Docusign Monitor herstellen
- Verbindung zu Domo herstellen
- Verbindung zu Dynatrace herstellen
- Verbindung zu Facebook-Werbeanzeigen herstellen
- Verbindung zu Facebook Page Insights herstellen
- Verbindung zu Freshdesk herstellen
- Verbindung zu Freshsales herstellen
- Verbindung zu Google Ads herstellen
- Verbindung zu Google Analytics herstellen 4
- Verbindung zu Google herstellen BigQuery in AWS Glue Studio
- Verbindung zur Google Search Console herstellen
- Verbindung zu Google Sheets herstellen

- Verbindung herstellen zu HubSpot
- Verbindung zu Instagram-Anzeigen herstellen
- Verbindung zur Gegensprechanlage herstellen in AWS Glue Studio
- Verbindung zu Jira Cloud herstellen
- Verbindung zu Kustomer herstellen
- Verbindung herstellen zu LinkedIn
- Verbindung zu Mailchimp herstellen
- Verbindung zu Microsoft Dynamics 365 CRM herstellen
- Verbindung zu Microsoft Teams herstellen
- Verbindung zu Mixpanel herstellen
- Verbindung zum Montag herstellen
- Verbindung zu MongoDB herstellen in AWS Glue Studio
- Verbindung zu Oracle herstellen NetSuite
- Verbindung zum OpenSearch Service herstellen in AWS Glue Studio
- Verbindung zu Okta herstellen
- Verbindung herstellen zu PayPal
- Verbindung zu Pendo herstellen
- Verbindung zu Pipedrive herstellen
- Verbindung zum Productboard herstellen
- Verbindung herstellen zu QuickBooks
- Verbindung zu Salesforce herstellen
- Verbindung zur Salesforce Marketing Cloud herstellen
- Verbindung zur Salesforce Commerce Cloud herstellen
- Verbindung mit Salesforce Marketing Cloud Account Engagement herstellen
- Verbindung zu SAP HANA herstellen in AWS Glue Studio
- Verbindung zu SAP herstellen OData
- Verbindung herstellen zu SendGrid
- Verbindung herstellen zu ServiceNow
- Verbindung zu Slack herstellen in AWS Glue Studio

- Verbindung zu Smartsheet herstellen
- Verbindung zu Snapchat-Anzeigen herstellen in AWS Glue Studio
- Verbindung zu Snowflake herstellen in AWS Glue Studio
- Verbindung zu Stripe herstellen in AWS Glue Studio
- Verbindung zu Teradata Vantage herstellen in AWS Glue Studio
- Verbindung zu Twilio herstellen
- Verbindung zu Vertica herstellen in AWS Glue Studio
- Verbindung herstellen zu WooCommerce
- Verbindung zu Zendesk herstellen
- Verbindung zu Zoho CRM herstellen
- Verbindung zu Zoom Meetings herstellen
- Hinzufügen einer JDBC-Verbindung mit Ihren eigenen JDBC-Treibern

Verbindung zu Adobe Analytics herstellen

Adobe Analytics ist eine robuste Datenanalyseplattform, die Daten aus digitalen Multichannel-Erlebnissen sammelt, die die Kundenreise unterstützen, und Tools zur Analyse der Daten bereitstellt. Es ist eine Plattform, die häufig von Marketern und Geschäftsanalysten für Geschäftsanalysezwecke verwendet wird. Wenn Sie ein Adobe Analytics-Benutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Adobe Analytics-Konto herstellen. Anschließend können Sie Adobe Analytics als Datenquelle in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Adobe Analytics und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Adobe Analytics
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Adobe Analytics
- Konfiguration von Adobe Analytics-Verbindungen
- Aus Adobe Analytics-Entitäten lesen
- Verbindungsoptionen für Adobe Analytics
- Ein Adobe Analytics-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Adobe Analytics

AWS Glue unterstützt Adobe Analytics wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Adobe Analytics abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Adobe Analytics-API-Versionen

v2.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Adobe Analytics

Bevor Sie die Übertragung von Adobe Analytics verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Adobe Analytics-Konto mit E-Mail-Adresse und Passwort. Weitere Informationen zum Erstellen eines Kontos finden Sie unter Erstellen eines Adobe Analytics-Kontos.
- Ihr Adobe Analytics-Konto ist für den API-Zugriff aktiviert. Der API-Zugriff ist standardmäßig für die Editionen Select, Prime und Ultimate aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Adobe Analytics-Konto herstellen. Für typische Verbindungen müssen Sie in Adobe Analytics nichts weiter tun.

Konfiguration von Adobe Analytics-Verbindungen

Adobe Analytics unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für`OAuth2`.

Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, `OAuth` da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet

werden. Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Adobe Analytics zu erstellen und beim Herstellen von Verbindungen über die Konsole ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben. AWS Glue In diesem Szenario werden sie weiterhin zu Adobe Analytics weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.

Eine öffentliche Adobe Analytics-Dokumentation zur Erstellung einer verbundenen App für den OAuth AUTHORIZATION_CODE-Flow finden Sie unter Adobe Analytics. APIs

So konfigurieren Sie eine Adobe Analytics-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:

   Für vom Kunden verwaltete verbundene App — Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, pro Verbindung ein Geheimnis zu erstellen AWS Glue.

2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Adobe Analytics aus.
   b. Geben Sie den `x_api_key, instanceUrl` Adobe Analytics-Dienst an, mit dem Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
    "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "secretsmanager:DescribeSecret",
```

```
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Adobe Analytics-Entitäten lesen

Voraussetzungen

Ein Adobe Analytics-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Anmerkung	Ja	Ja	Ja	Ja	Nein
Berechnete Metriken	Ja	Ja	Ja	Ja	Nein
Funktion für berechnete Metriken	Ja	Nein	Nein	Ja	Nein
Geteilte Metadaten	Ja	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
von Komponenten					
Datumsbereiche	Ja	Ja	Nein	Ja	Nein
Dimensionen	Ja	Nein	Nein	Ja	Nein
Metriken	Ja	Nein	Nein	Ja	Nein
Projekte	Ja	Nein	Nein	Ja	Nein
Meldet den wichtigsten Artikel	Ja	Ja	Nein	Ja	Nein
Segmente	Ja	Ja	Ja	Ja	Nein
Nutzungsprotokolle	Ja	Ja	Nein	Ja	Nein

Beispiel

```
adobeAnalytics_read = glueContext.create_dynamic_frame.from_options(
     connection_type="adobeanalytics",
     connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "annotation/ex*****",
        "API_VERSION": "v2.0"
 })
```

Entitäts- und Felddetails von Adobe Analytics

- Anmerkungen
- Berechnete Metriken

- Metadaten der Komponenten
- Datumsbereiche
- Dimensions (Abmessungen)
- Metriken
- Projekte
- Berichte
- Segmente
- Benutzer
- Nutzungsprotokolle

Verbindungsoptionen für Adobe Analytics

Im Folgenden sind die Verbindungsoptionen für Adobe Analytics aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Adobe Analytics.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Die Adobe Analytics-Rest-API-Version, die Sie verwenden möchten. Zum Beispiel: v2.0.
- `X_API_KEY`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Es ist erforderlich, um den Entwickler oder die Anwendung zu authentifizieren, die Anfragen an die API stellen.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Ein Adobe Analytics-Konto erstellen

1. Registrieren Sie sich für das Austauschpartnerprogramm, indem Sie auf das Adobe-Partnerprogramm zugreifen.
2. Wählen Sie „Am Exchange-Programm teilnehmen“.
3. Registrieren Sie sich oder erstellen Sie ein Konto mit Ihrer Unternehmens-E-Mail-Adresse.

4. Wählen Sie im Vorschlagsfeld das entsprechende Unternehmen aus, das über ein Adobe Analytics-Produktabonnement verfügt.
5. Stellen Sie sicher, dass das Konto bei einer gültigen Organisation (aus der verfügbaren Liste) registriert ist, die über ein aktives Adobe Analytics-Abonnement verfügt.
6. Aktivieren Sie Ihr Konto nach der Genehmigung durch die Unternehmensverwaltung, indem Sie auf den Link in der Genehmigungs-E-Mail klicken.

Überprüfen Sie, ob das von Ihnen erstellte Konto Zugriff auf den Adobe Analytics-Dienst hat

1. Melden Sie sich bei der Adobe Admin Console an.
2. Überprüfen Sie den Namen der Organisation in der oberen rechten Ecke der Seite, um sicherzustellen, dass Sie sich bei der richtigen Firma angemeldet haben.
3. Wählen Sie Produkte aus und überprüfen Sie, ob Adobe Analytics verfügbar ist.

> **Note**
>
> Wenn keine Organisation verfügbar ist oder das Adobe Analytics-Produkt ausgegraut oder nicht verfügbar ist, ist Ihr Konto wahrscheinlich keiner Organisation zugeordnet und/oder es gibt kein aktives Adobe Analytics-Abonnement. Wenden Sie sich an Ihren Systemadministrator, um den Zugriff auf diesen Service für Ihr Konto anzufordern.

Ein Projekt und **`OAuth2.0`** Anmeldeinformationen erstellen

1. Melden Sie sich bei dem Adobe Analytics-Konto an, in dem die OAuth 2.0-App erstellt werden soll.
2. Wählen Sie Projekt und dann Neues Projekt erstellen aus.
3. Um ein Projekt hinzuzufügen, wählen Sie Zum Projekt hinzufügen und dann API aus.
4. Wählen Sie Adobe Analytics API aus.
5. Wählen Sie OAUTH als Benutzerauthentifizierung aus.
6. Wählen Sie Web as aus `0AUTH` und geben Sie die Umleitungs-URI an.

   Informationen zur Umleitungs-URI und ihrem Muster finden Sie im Folgenden:

- `OAuth 2.0`Standard-Umleitungs-URI — Ein Standard-Umleitungs-URI ist die URL der Seite, auf die Adobe während des Authentifizierungsprozesses zugreift. Beispiel: `https://ap-southeast-2.console.aws.amazon.com/appflow/oauth`
- OAuth 2.0 Umleitungs-URI-Muster — Ein Umleitungs-URI-Muster ist ein URI-Pfad (oder eine durch Kommas getrennte Liste von Pfaden), zu dem Adobe (falls angefordert) weiterleiten kann, wenn der Anmeldevorgang abgeschlossen ist. Beispiel: `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`

7. Fügen Sie die folgenden Bereiche hinzu:
   - `openid`
   - `read_organizations`
   - `additional_info.projectedProductContext`
   - `additional_info.job_function`
8. Wählen Sie Anmeldeinformationen speichern aus.
9. Nachdem die App erstellt wurde, kopieren Sie die `Client Secret` Werte `Client ID` und in eine Textdatei.

Einschränkungen

Die folgenden Einschränkungen gelten für den Adobe Analytics-Connector:

- Adobe Analytics unterstützt keine feld- und datensatzbasierte Partitionierung. Feldbasierte Partitionierung wird nicht unterstützt, da Sie Felder, die Sie partitionieren, nicht abfragen können. Die datensatzbasierte Partitionierung kann nicht unterstützt werden, da es keine Möglichkeit gibt, einen Offset für die Paginierung zu erhalten.
- In der `Report Top Item` Entität funktionieren die Abfrageparameter `startDate` und die `endDate` Abfrageparameter nicht wie erwartet. Die Antwort wird nicht auf der Grundlage dieser Parameter gefiltert, was zu Problemen mit dem Filter und dem inkrementellen Fluss für diese Entität führt.
- Für die `Segment` Entitäten `Annotation` `Calculated MetricsCalculated Metrics Function`,`Date Ranges`,`Dimension`,`Metric`,, `ProjectReport Top Items`, und gibt der `locale` Abfrageparameter an, welche Sprache für lokalisierte Antwortabschnitte verwendet werden soll, und filtert die Datensätze nicht. Zeigt die `locale="ja_JP"` Daten beispielsweise auf Japanisch an.

- `Report Top Item`Entität — Filter aktiviert `dateRange` und `lookupNoneValues` Felder funktionieren derzeit nicht.
- `Segment`Entität: Mit einem Filterwert `includeType="templates"` funktionieren Filter für andere Felder nicht.
- `Date Range`Entität — der Filter für das `curatedRsid` Feld funktioniert nicht.
- `Metric entity`Entität — Der Filter nach einem segmentierbaren Feld mit dem Wert „falsch“ liefert Ergebnisse sowohl für den wahren als auch für den falschen Wert.

Verbindung zu Adobe Marketo Engage herstellen

Adobe Marketo Engage ist eine Plattform zur Marketingautomatisierung, die es Marketingexperten ermöglicht, personalisierte Multi-Channel-Programme und -Kampagnen für Interessenten und Kunden zu verwalten.

Themen

- AWS Glue Unterstützung für Adobe Marketo Engage
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Adobe Marketo Engage
- Konfiguration von Adobe Marketo Engage-Verbindungen
- Lesen aus Adobe Marketo Engage-Entitäten
- Verbindungsoptionen für Adobe Marketo Engage
- Einschränkungen und Hinweise für den Adobe Marketo Engage Connector

AWS Glue Unterstützung für Adobe Marketo Engage

AWS Glue unterstützt Adobe Marketo Engage wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Adobe Marketo Engage abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte Adobe Marketo Engage API-Versionen

Die folgenden Adobe Marketo Engage API-Versionen werden unterstützt:

- v1

Informationen zur versionsspezifischen Unterstützung von Entitäten finden Sie unter Unterstützte Entitäten für Source.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Adobe Marketo Engage

Bevor Sie Daten von Adobe AWS Glue Marketo Engage übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Die folgenden Mindestanforderungen sind:

- Sie haben ein Adobe Marketo Engage-Konto mit Kundenanmeldedaten.
- Ihr Adobe Marketo Engage-Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung zu Ihrem Adobe AWS Glue Marketo Engage-Konto herstellen. Für typische Verbindungen müssen Sie in Adobe Marketo Engage nichts weiter tun.

Anmeldeinformationen für Version 2.0 erhalten OAuth

Informationen zum Abrufen von API-Anmeldeinformationen, mit denen Sie authentifizierte Aufrufe an Ihre Instanz tätigen können, finden Sie unter REST-API im Adobe Marketo Engage Developer Guide.

Konfiguration von Adobe Marketo Engage-Verbindungen

Adobe Marketo Engage unterstützt den Gewährungstyp CLIENT CREDENTIALS für. OAuth2

- Dieser Zuschusstyp wird als 2-Legged OAuth 2.0 betrachtet, da er von Kunden verwendet wird, um außerhalb des Benutzerkontextes ein Zugriffstoken zu erhalten. AWS Glue ist in der Lage, die Client-ID und das Client-Geheimnis zur Authentifizierung von Adobe Marketo Engage zu verwenden, die von benutzerdefinierten Diensten bereitgestellt werden APIs , die Sie definieren.
- Jeder benutzerdefinierte Dienst gehört einem Benutzer, der nur über eine API verfügt und über eine Reihe von Rollen und Berechtigungen verfügt, die den Dienst autorisieren, bestimmte

Aktionen auszuführen. Ein Zugriffstoken ist einem einzelnen benutzerdefinierten Dienst zugeordnet.

- Dieser Gewährungstyp führt zu einem Zugriffstoken, das nur von kurzer Dauer ist und durch Aufrufen eines Identitätsendpunkts erneuert werden kann.
- Öffentliche Adobe Marketo Engage-Dokumentation für OAuth 2.0 mit Kundenanmeldedaten finden Sie unter Authentifizierung im Adobe Marketo Engage Developer Guide.

So konfigurieren Sie eine Adobe Marketo Engage-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen für jede Verbindung in ein Secret erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Adobe Marketo Engage.
   b. Geben Sie die `INSTANCE_URL` Adobe Marketo Engage-Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen aus Adobe Marketo Engage-Entitäten

Voraussetzung

Ein Adobe Marketo Engage-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen, z. B. Leads oder Aktivitäten oder benutzerdefinierte Objekte. Die folgenden Tabellen zeigen die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle (synchron):

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
führt	Ja	Ja	Nein	Ja	Nein
Aktivitäten	Ja	Ja	Nein	Ja	Nein
benutzerdefinierte Objekte	Ja	Ja	Nein	Ja	Nein

Unterstützte Entitäten für die Quelle (asynchron):

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
führt	Ja	Nein	Nein	Ja	Ja
Aktivitäten	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
benutzerdefinierte Objekte	Ja	Nein	Nein	Ja	Ja

Beispiel:

```
adobe-marketo-engage_read = glueContext.create_dynamic_frame.from_options(
    connection_type="adobe-marketo-engage",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://539-t**-6**.mktorest.com"
    }
```

Entitäts- und Felddetails von Adobe Marketo Engage:

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Aktivitäten	sinceDateTime (wird nur synchron unterstützt)	DateTime	>= (nur für den synchronen Modus)
	createdAt (nur asynchron unterstützt)	DateTime	zwischen (nur für den asynchronen Modus)
	activitiesTypeId	Ganzzahl	=
	adobe-marketo-engageGUID	Long	= (nur für den synchronen Modus)
	LeadID	Long	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Datum der Aktivität	DateTime	N/A
	campaignId	Long	N/A
	primaryAttributeValueID	Ganzzahl	N/A
	primaryAttributeValue	String	N/A
	Attribute	String	N/A

Entitäten mit dynamischen Metadaten:

Für die folgenden Entitäten bietet Adobe Marketo Engage Endpunkte zum dynamischen Abrufen von Metadaten, sodass die Operatorunterstützung für jede Entität auf Datentypebene erfasst wird.

Entität	Datentyp	Unterstützte Operatoren
führt	Ganzzahl	= (nur für den synchronen Modus)
	DateTime	zwischen (nur für den asynchronen Modus)
	String	= (nur für den synchronen Modus)
	Long	N/A
	Boolesch	N/A
	Datum	N/A
	Gleitkommazahl	N/A
benutzerdefinierte Objekte	Ganzzahl	N/A

Entität	Datentyp	Unterstützte Operatoren
	DateTime	zwischen (nur für den asynchronen Modus)
	String	= (nur für den synchronen Modus)
	Datum	N/A
	Long	N/A
	Boolesch	N/A
	Gleitkommazahl	N/A

Abfragen partitionieren

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2024-07-01T00:00:00.000Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

In der folgenden Tabelle werden die Informationen zur Unterstützung von Feldern zur Partitionierung von Entitäten beschrieben:

Entity-Name	Partitionierung von Feldern	Datentyp
führt	createdAt	DateTime
	Aktualisieren bei	DateTime
benutzerdefinierte Objekte	updatedAt	DateTime

Beispiel:

```
adobe-marketo-engage_read = glueContext.create_dynamic_frame.from_options(
    connection_type="adobe-marketo-engage",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2024-07-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-07-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Adobe Marketo Engage

Im Folgenden sind die Verbindungsoptionen für Adobe Marketo Engage aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Adobe Marketo Engage.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Version der Adobe Marketo Engage Rest-API, die Sie verwenden möchten. Zum Beispiel: v1.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.

- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.
- `TRANSFER_MODE`(Zeichenfolge) — Standard: SYNC. Wird für asynchrones Lesen verwendet.

Einschränkungen und Hinweise für den Adobe Marketo Engage Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Adobe Marketo Engage-Connector:

- 'SinceDateTime' und 'activityTypeId' sind obligatorische Filterparameter für die Entität „Synchronisierungsaktivitäten“.
- Den Abonnements werden 50.000 API-Aufrufe pro Tag zugewiesen (was täglich um 12:00 Uhr CST zurückgesetzt wird). Zusätzliche Tageskapazität kann im Rahmen eines Adobe Marketo Engage-Abonnements erworben werden.
- Die maximale Zeitspanne für den Datumsbereichsfilter (`createdAt`oder`updatedAt`) beträgt 31 Tage.
- Den Abonnements werden jeweils maximal 10 Massenextraktionsaufträge in der Warteschlange zugewiesen.
- Standardmäßig sind Extrahierungsaufträge auf 500 MB pro Tag begrenzt (was täglich um 12:00 Uhr CST zurückgesetzt wird). Zusätzliche Tageskapazität kann im Rahmen eines Adobe Marketo Engage-Abonnements erworben werden.
- Die maximale Anzahl gleichzeitiger Exportaufträge ist 2.
- Die maximale Anzahl von Exportaufträgen in der Warteschlange (einschließlich der aktuell exportierenden Jobs) beträgt 10.
- Die maximal zulässige Dateigröße für das Extrahieren aus einem Bulk-Job beträgt 1 GB.
- Sobald ein asynchroner Auftrag erstellt wurde, läuft die Aufbewahrungsfrist für Dateien 7 Tage ab.
- `createdAt`oder `updatedAt` sind obligatorische Filterparameter für die Async Leads-Entität.
- `createdAt`ist ein obligatorischer Filterparameter für die Entität Async Activities.
- `updatedAt`ist ein obligatorischer Filterparameter für die Async Custom Object-Entität.

Weitere Informationen finden Sie unter Bewährte Methoden für die Integration von Adobe Marketo Engage und Bulk Extract.

Verbindung zu Amazon Redshift herstellen in AWS Glue Studio

> **Note**
>
> Sie können Folgendes verwenden … AWS Glue für Spark, um aus Tabellen in Amazon Redshift Datenbanken außerhalb von zu lesen und in sie zu schreiben AWS Glue Studio. Zur Konfiguration Amazon Redshift mit AWS Glue Jobs programmgesteuert, sieheRedshift-Verbindungen.

AWS Glue bietet integrierte Unterstützung für Amazon Redshift. AWS Glue Studio bietet eine visuelle Oberfläche, mit der Sie eine Verbindung herstellen Amazon Redshift, Datenintegrationsjobs erstellen und auf ihnen ausführen können AWS Glue Studio Serverlose Spark-Laufzeit.

Themen

- Eine Amazon Redshift Verbindung herstellen
- Einen Amazon Redshift Quellknoten erstellen
- Einen Amazon Redshift Zielknoten erstellen
- Erweiterte Optionen

Eine Amazon Redshift Verbindung herstellen

Berechtigungen erforderlich

Für die Verwendung von Amazon Redshift Clustern und Amazon Redshift serverlosen Umgebungen sind zusätzliche Berechtigungen erforderlich. Weitere Informationen zum Hinzufügen von Berechtigungen zu ETL-Aufträgen finden Sie unter Überprüfen der für ETL-Aufträge erforderlichen IAM-Berechtigungen.

- Rotverschiebung: DescribeClusters
- Redshift — serverlos: ListWorkgroups
- redshift-serverlos: ListNamespaces

Übersicht

Beim Hinzufügen einer Amazon Redshift Verbindung können Sie eine bestehende Amazon Redshift Verbindung auswählen oder eine neue Verbindung erstellen, wenn Sie einen Datenquellen-Redshift-Knoten hinzufügen AWS Glue Studio.

AWS Glue unterstützt sowohl Amazon Redshift Cluster als auch Amazon Redshift serverlose Umgebungen. Wenn Sie eine Verbindung herstellen, wird in Amazon Redshift serverlosen Umgebungen die Bezeichnung Serverless neben der Verbindungsoption angezeigt.

Weitere Informationen zum Herstellen einer Amazon Redshift Verbindung finden Sie unter Daten hin und her verschieben. Amazon Redshift

Einen Amazon Redshift Quellknoten erstellen

Berechtigungen erforderlich

AWS Glue Studio Jobs, die Amazon Redshift Datenquellen verwenden, erfordern zusätzliche Berechtigungen. Weitere Informationen zum Hinzufügen von Berechtigungen zu ETL-Aufträgen finden Sie unter Überprüfen der für ETL-Aufträge erforderlichen IAM-Berechtigungen.

Die folgenden Berechtigungen sind erforderlich, um eine Amazon Redshift Verbindung verwenden zu können.

- Redshift-Daten: ListSchemas
- Redshift-Daten: ListTables
- Redshift-Daten: DescribeTable
- Redshift-Daten: ExecuteStatement
- Redshift-Daten: DescribeStatement
- Redshift-Daten: GetStatementResult

Eine Datenquelle hinzufügen Amazon Redshift

So fügen Sie einen Knoten von Datenquelle – Amazon Redshift hinzu:

1. Wählen Sie den Amazon Redshift Zugriffstyp:

    - Direkte Datenverbindung (empfohlen) – wählen Sie diese Option, wenn Sie direkt auf Ihre Amazon Redshift -Daten zugreifen möchten. Dies ist die empfohlene Option und auch die Standardeinstellung.

- Data Catalog tables — Wählen Sie diese Option, wenn Sie Datenkatalogtabellen haben, die Sie verwenden möchten.

2. Wenn Sie Direkte Datenverbindung wählen, wählen Sie die Verbindung für Ihre Amazon Redshift Datenquelle. Dabei wird davon ausgegangen, dass die Verbindung bereits besteht und Sie aus bestehenden Verbindungen auswählen können. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Redshift-Verbindung erstellen. Weitere Informationen finden Sie unter Übersicht über die Verwendung von Konnektoren und Verbindungen.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken. Informationen zur Verbindung sind sichtbar, einschließlich URL, Sicherheitsgruppen, Subnetz, Verfügbarkeitszone, Beschreibung sowie erstellte (UTC) und letzte aktualisierte (UTC) Zeitstempel.

3. Wählen Sie eine Amazon Redshift Quelloption:

   - Wählen Sie eine einzelne Tabelle — das ist die Tabelle, die die Daten enthält, auf die Sie von einer einzigen Amazon Redshift Tabelle aus zugreifen möchten.
   - Benutzerdefinierte Abfrage eingeben – ermöglicht Ihnen den Zugriff auf einen Datensatz aus mehreren Amazon Redshift -Tabellen basierend auf Ihrer benutzerdefinierten Abfrage.

4. Wenn Sie eine einzelne Tabelle ausgewählt haben, wählen Sie das Amazon Redshift Schema. Die Liste der verfügbaren Schemas zur Auswahl wird durch die ausgewählte Tabelle bestimmt.

   Oder wählen Sie Benutzerdefinierte Abfrage eingeben aus. Wählen Sie diese Option, um aus mehreren Amazon Redshift -Tabellen auf einen benutzerdefinierten Datensatz zuzugreifen. Wenn Sie diese Option wählen, geben Sie die Amazon Redshift Abfrage ein.

   Wenn Sie eine Verbindung zu einer Amazon Redshift serverlosen Umgebung herstellen, fügen Sie der benutzerdefinierten Abfrage die folgende Berechtigung hinzu:

   ```
   GRANT SELECT ON ALL TABLES IN <schema> TO PUBLIC
   ```

   Sie können Schema ableiten auswählen, um das Schema basierend auf der von Ihnen eingegebenen Abfrage zu lesen. Sie können auch Redshift Query Editor öffnen wählen, um eine Amazon Redshift Abfrage einzugeben. Weitere Informationen finden Sie unter Abfragen einer Datenbank mit dem Abfrage-Editor.

5. Wählen Sie unter Leistung und Sicherheit das Amazon-S3-Staging-Verzeichnis und die IAM-Rolle aus.

   - Amazon-S3-Staging-Verzeichnis – Wählen Sie den Amazon-S3-Speicherort für die temporäre Bereitstellung von Daten.
   - IAM-Rolle – Wählen Sie die IAM-Rolle aus, die an den von Ihnen ausgewählten Amazon-S3-Speicherort schreiben kann.

6. Geben Sie unter Benutzerdefinierte Redshift-Parameter – optional den Parameter und den Wert ein.

Einen Amazon Redshift Zielknoten erstellen

Berechtigungen erforderlich

AWS Glue Studio Jobs, die Amazon Redshift Data Target verwenden, erfordern zusätzliche Berechtigungen. Weitere Informationen zum Hinzufügen von Berechtigungen zu ETL-Aufträgen finden Sie unter Überprüfen der für ETL-Aufträge erforderlichen IAM-Berechtigungen.

Die folgenden Berechtigungen sind erforderlich, um eine Amazon Redshift Verbindung verwenden zu können.

- Redshift-Daten: ListSchemas
- Redshift-Daten: ListTables

Einen Zielknoten hinzufügen Amazon Redshift

Um einen Amazon Redshift Zielknoten zu erstellen:

1. Wählen Sie eine vorhandene Amazon Redshift Tabelle als Ziel aus, oder geben Sie einen neuen Tabellennamen ein.
2. Wenn Sie den Zielknoten Datenziel – Redshift verwenden, können Sie aus den folgenden Optionen auswählen:

   - APPEND – Wenn eine Tabelle bereits vorhanden ist, übertragen Sie alle neuen Daten als Einfügung in diese Tabelle. Wenn die Tabelle nicht vorhanden ist, erstellen Sie sie und fügen Sie dann alle neuen Daten ein.

Aktivieren Sie außerdem das Kontrollkästchen, wenn Sie vorhandene Datensätze in der Zieltabelle aktualisieren (UPSERT) möchten. Die Tabelle muss zuerst vorhanden sein, andernfalls schlägt der Vorgang fehl.

- ZUSAMMENFÜHREN — AWS Glue aktualisiert Ihre Zieltabelle auf der Grundlage der von Ihnen angegebenen Bedingungen oder fügt Daten an sie an.

  > **Note**
  >
  > Um die Zusammenführungsaktion zu verwenden in AWS Glue, müssen Sie die Amazon Redshift Zusammenführungsfunktion aktivieren. Anweisungen zum Aktivieren der Zusammenführung für Ihre Amazon Redshift Instance finden Sie unter MERGE (Vorschau).

  Wählen Sie die Optionen aus:

  - Schlüssel und einfache Aktionen auswählen – wählen Sie die Spalten, die als übereinstimmende Schlüssel zwischen den Quelldaten und Ihrem Zieldatensatz verwendet werden sollen.

    Geben Sie bei Übereinstimmung die folgenden Optionen an:

    - Aktualisieren Sie den Datensatz in Ihrem Zieldatensatz mit Daten aus der Quelle.
    - Löschen Sie den Datensatz in Ihrem Zieldatensatz.

    Geben Sie die folgenden Optionen an, wenn keine Übereinstimmung vorliegt:

    - Fügen Sie Quelldaten als neue Zeile in Ihren Zieldatensatz ein.
    - Nichts unternehmen.
  - Eine benutzerdefinierte MERGE-Anweisung eingeben – Sie können dann Zusammenführungs-Anweisung validieren auswählen, um zu überprüfen, ob die Anweisung gültig oder ungültig ist.
- TRUNCATE – Wenn bereits eine Tabelle vorhanden ist, kürzen Sie die Tabellendaten, indem Sie zunächst den Inhalt der Zieltabelle löschen. Wenn das Kürzen erfolgreich ist, fügen Sie alle Daten ein. Wenn die Tabelle nicht vorhanden ist, erstellen Sie die Tabelle und fügen Sie alle Daten ein. Wenn das Kürzen nicht erfolgreich ist, schlägt der Vorgang fehl.
- DROP – Wenn eine Tabelle bereits vorhanden ist, löschen Sie die Tabellenmetadaten und -daten. Wenn der Löschvorgang erfolgreich ist, fügen Sie alle Daten ein. Wenn die Tabelle

nicht vorhanden ist, erstellen Sie die Tabelle und fügen Sie alle Daten ein. Wenn das Löschen nicht erfolgreich ist, schlägt der Vorgang fehl.

- CREATE – Erstellen Sie eine neue Tabelle mit dem Standardnamen. Wenn der Tabellenname bereits vorhanden ist, erstellen Sie aus Gründen der Eindeutigkeit eine neue Tabelle mit dem Namenszusatz von `job_datetime`. Dadurch werden alle Daten in die neue Tabelle eingefügt. Wenn die Tabelle vorhanden ist, wird an den endgültigen Tabellennamen der Nachsatz angefügt. Wenn die Tabelle nicht vorhanden ist, wird eine Tabelle erstellt. In beiden Fällen wird eine neue Tabelle erstellt.

Erweiterte Optionen

Siehe Verwenden des Amazon Redshift Spark-Connectors auf AWS Glue.

Verbindung zu Asana herstellen

Asana ist eine cloudbasierte Lösung für die Teamzusammenarbeit, die Teams dabei unterstützt, Aufgaben und Projekte zu organisieren, zu planen und abzuschließen. Wenn Sie ein Asana-Nutzer sind, enthält Ihr Konto Daten über Ihre Arbeitsbereiche, Projekte, Aufgaben, Teams und mehr. Sie können Daten von Asana an bestimmte AWS Dienste oder andere unterstützte Anwendungen übertragen.

Themen

- AWS Glue Unterstützung für Asana
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Asana konfigurieren
- Asana-Verbindungen konfigurieren
- Aus Asana-Entitäten lesen
- Asana-Verbindungsoptionen
- Ein Asana-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Asana

AWS Glue unterstützt Asana wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Asana abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Asana-API-Versionen

1,0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Asana konfigurieren

Bevor Sie AWS Glue den Transfer von Asana nutzen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Asana-Konto mit E-Mail und Passwort. Weitere Informationen zur Erstellung eines Kontos finden Sie unter Ein Asana-Konto erstellen.
- Sie müssen AWS ein Konto mit dem Servicezugriff auf AWS Glue erstellt haben.
- Stellen Sie sicher, dass Sie eine der folgenden Ressourcen in Ihrem Asana-Konto erstellt haben:
  - Eine Entwickler-App, die `OAuth 2.0` Authentifizierung unterstützt. Weitere Anweisungen finden Sie OAuthin der Asana-Dokumentation für Entwickler. Eine alternative Vorgehensweise finden Sie unter the section called “Ein Asana-Konto erstellen”.
  - Ein persönliches Zugriffstoken. Weitere Informationen finden Sie unter dem persönlichen Zugriffstoken https://developers.asana.com/docs/personal-access-token in der Asana Developers-Dokumentation.

Wenn Sie diese Anforderungen erfüllen, sind Sie bereit, eine Verbindung AWS Glue zu Ihrem Adobe Analytics-Konto herzustellen. Für typische Verbindungen müssen Sie in Adobe Analytics nichts weiter tun.

Asana-Verbindungen konfigurieren

Asana unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für. `OAuth2`

Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, `0Auth` da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Asana zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Asana weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Diese Art der Gewährung führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.

Eine öffentliche Asana-Dokumentation zur Erstellung einer verbundenen App für `AUTHORIZATION_CODE 0Auth` Flow finden Sie unter Asana. APIs

So konfigurieren Sie eine Asana-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Für vom Kunden verwaltete verbundene App — Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   -
     > **Note**
     >
     > Sie müssen ein Geheimnis für die Verbindung in erstellen AWS Glue.
2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Asana aus.
   b. Stellen Sie die Asana-Umgebung bereit.
   c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
```

```
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Asana-Entitäten lesen

Voraussetzungen

Ein Asana-Objekt, aus dem Sie gerne lesen würden. Sehen Sie sich die Tabelle mit den unterstützten Entitäten unten an, um die verfügbaren Entitäten zu überprüfen.

Quelle: Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Workspace	Nein	Ja	Nein	Ja	Nein
Markierung	Nein	Ja	Nein	Ja	Nein
Benutzer	Nein	Ja	Nein	Ja	Nein
Portfolio	Nein	Ja	Nein	Ja	Nein
Team	Nein	Ja	Nein	Ja	Nein
Projekt	Ja	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Abschnitt	Nein	Ja	Nein	Ja	Nein
Aufgabe	Ja	Nein	Nein	Ja	Ja
Ziel	Ja	Ja	Nein	Ja	Nein
AuditLogEvent	Ja	Ja	Nein	Ja	Nein
Status-Update	Ja	Ja	Nein	Ja	Nein
Benutzerdefiniertes Feld	Nein	Ja	Nein	Ja	Nein
Kurzbeschreibung des Projekts	Ja	Nein	Nein	Ja	Ja

Beispiel

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Asana",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "task/workspace:xxxx",
        "API_VERSION": "1.0",
        "PARTITION_FIELD": "created_at",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Angaben zu Entität und Feld in Asana

- Workspace
- Markierung
- Nutzer
- Portfolio
- Mannschaft
- Projekt
- Abschnitt
- Aufgabe
- Ziel
- AuditLogEvent
- Status-Update
- Benutzerdefiniertes Feld
- Kurzbeschreibung des Projekts

Abfragen partitionieren

Zusätzliche Spark-OptionenPARTITION_FIELD,LOWER_BOUND,UPPER_BOUND, NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:2024-06-07T13:30:00.134Z.
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt.

Entity Name	Feld partitionieren	Datentyp
Aufgabe	created_at	DateTime
Aufgabe	modified_at	DateTime

Beispiel

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Asana",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "task/workspace:xxxx",
        "API_VERSION": "1.0",
        "PARTITION_FIELD": "created_at",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Asana-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Asana aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Asana.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Asana Rest API-Version, die Sie verwenden möchten. Zum Beispiel: 1.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.

- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Ein Asana-Konto erstellen

1. Eröffnen Sie ein Asana-Konto und wählen Sie Registrieren.
2. Nachdem Sie sich angemeldet haben, werden Sie zur Seite zur Kontoeinrichtung weitergeleitet. Führen Sie folgende Schritte aus:
   - Überprüfen Sie das Formular zur Kontoeinrichtung.
   - Füllen Sie alle relevanten Details aus, um Ihr Asana-Konto zu erstellen.
   - Überprüfen Sie die Informationen noch einmal auf ihre Richtigkeit.
3. Wählen Sie „Konto erstellen“ oder „Senden“ (der genaue Text der Schaltfläche kann variieren), um die Einrichtung Ihres Kontos abzuschließen.

Die App in Asana erstellen für **`OAuth2.0`**

1. Melden Sie sich mit Ihren Asana-Kundendaten beim Asana-Konto an.
2. Wählen Sie Ihr Benutzerprofilsymbol in der oberen rechten Ecke und wählen Sie Meine Einstellungen aus dem Drop-down-Menü aus.
3. Wählen Sie den Tab Apps und dann Entwickler-Apps verwalten aus.
4. Wählen Sie Neue App erstellen aus und geben Sie die entsprechenden Details ein.
5. Wählen Sie Apps erstellen.
6. Auf der Seite Meine Apps:
   a. Wählen Sie aus OAuthund notieren Sie sich im Abschnitt App-Anmeldeinformationen Ihre Client-ID und Ihren geheimen Kundenschlüssel.
   b. Fügen Sie im URLs Abschnitt Weiterleitung die erforderlichen Umleitungs-URLs hinzu.

> **Note**
>
> Geben Sie die Umleitungs-URI in diesem Format ein:`https://{aws-region-code}.console.aws.amazon.com/gluestudio/oauth`.
> Beispiel: Verwenden Sie für den Osten der USA (Nord-Virginia): `https://us-east-1.console.aws.amazon.com/gluestudio/oauth`

Die App in Asana für **`PAT`** Token erstellen

1. Melden Sie sich mit Ihren Asana-Kundendaten beim Asana-Konto an.
2. Wählen Sie auf Ihr Benutzerprofilsymbol in der oberen rechten Ecke und wählen Sie Meine Profileinstellungen aus dem Drop-down-Menü aus.
3. Wählen Sie den Tab Apps und dann Dienstkonten aus.
4. Wählen Sie Neue App erstellen aus und geben Sie die entsprechenden Details ein.
5. Wählen Sie Dienstkonto hinzufügen aus.
6. Auf der nächsten Seite wird Ihr Token angezeigt. Kopieren Sie Ihr Token und speichern Sie es sicher.

> **Important**
>
> Dieses Token wird nur einmal angezeigt. Stellen Sie sicher, dass Sie es kopieren und sicher aufbewahren.

Einschränkungen

Die folgenden Einschränkungen gelten für den Asana-Connector:

- Dienstkonten in Unternehmensdomänen können nur auf API-Endpunkte für Audit-Logs zugreifen. Für den Zugriff auf diese Endpunkte ist eine Authentifizierung mit dem persönlichen Zugriffstoken eines Dienstkontos erforderlich.
- Auf die Goal-Entität kann nur für Benutzerkonten mit einem Premium-Plan oder höher zugegriffen werden.

- `Audit Log Event Entity`— Im Connector werden beide `end_at` Felder zu einem einzigen Feld „start_end_at“ zusammengefasst, um Filterung `start_at` und inkrementelle Übertragung zu unterstützen.
- Die Partitionierung für das `Date` Feld kann nicht unterstützt werden, obwohl es die Operatoren -to und -to unterstützt. greater-than-or-equal less-than-or-equal Szenario: Es wurde ein Job mit `partitionField` as `due_on` (Datentyp: Datum), `lowerBound` as`2019-09-14`, `upperBound` as und as erstellt. `2019-09-16` `numPartition` 2 Der Filterteil der Endpunkt-URL wird wie folgt erstellt:
  - Partition1: due_on.before=2019-09-14&due_on.after=2019-09-14
  - partition2: due_on.before=2019-09-15&due_on.after=2019-09-15 Ausgabe:
  - In Partition1 erhalten wir Daten mit due_date als 2019-09-14 und 2019-09-15
  - In Partition2 erhalten wir dieselben Daten mit due_date wie 2019-09-15 (die sich in Partition1 befanden) zusammen mit anderen Daten, was zu Datenduplikationen führt.
- Filterung und Partitionierung können nicht für dasselbe Feld unterstützt werden, da von der SaaS-Seite ein fehlerhafter Anforderungsfehler ausgelöst wird.
- Die Task-Entität erfordert mindestens ein Feld in den Filterkriterien. Bei Asana gibt es eine Einschränkung, bei der die Seitennummerierung nicht identifiziert wird, ohne die Datensätze anhand eines zeitbasierten Felds zu sortieren. Daher wird das Feld Created_AT zusammen mit der Paginierung verwendet, um die nächste Gruppe von Datensätzen zu unterscheiden. Das Feld Created_AT ist im Filter als Pflichtfeld markiert und hat den Standardwert 2000-01-01T 00:00:00 Z, falls es nicht angegeben wird. Weitere Informationen zur Paginierung finden Sie unter Aufgaben in einem Arbeitsbereich.

Verbindung zu Azure Cosmos DB herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für Azure Cosmos DB. AWS Glue Studio bietet eine visuelle Oberfläche, um eine Verbindung zu Azure Cosmos DB für NoSQL herzustellen, Datenintegrationsjobs zu erstellen und sie auf dem AWS Glue Studio serverlose Spark-Laufzeit.

Themen

- Erstellen einer Azure-Cosmos-DB-Verbindung
- Erstellen eines Azure-Cosmos-DB-Quellknotens
- Erstellen eines Azure-Cosmos-DB-Zielknotens
- Erweiterte Optionen

Erstellen einer Azure-Cosmos-DB-Verbindung

Voraussetzungen:

- In Azure müssen Sie einen Azure Cosmos DB-Schlüssel für die Verwendung durch AWS Glue, `cosmosKey` identifizieren oder generieren. Weitere Informationen finden Sie unter Sicherer Zugriff auf Daten in Azure Cosmos DB in der Azure-Dokumentation.

Eine Verbindung zu Azure Cosmos DB konfigurieren Sie wie folgt:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit Ihrem Azure Cosmos DB-Schlüssel. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `spark.cosmos.accountKey` mit dem Wert. *`cosmosKey`*
2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen,*`connectionName`*, für die future Verwendung in AWS Glue.
   - Wählen Sie als Verbindungstyp Azure Cosmos DB aus.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*`secretName`*.

Erstellen eines Azure-Cosmos-DB-Quellknotens

Voraussetzungen

- Eine AWS Glue Azure Cosmos DB-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Azure-Cosmos-DB-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Ein Container in Azure Cosmos DB für NoSQL, aus dem Sie lesen möchten. Sie benötigen Identifikationsinformationen für den Container.

Ein Container in Azure Cosmos DB für NoSQL wird anhand seiner Datenbank und seines Containers identifiziert. Sie müssen die Datenbank- und Container-Namen angeben*cosmosContainerName*, wenn Sie eine Verbindung zur Azure Cosmos for NoSQL-API herstellen. *cosmosDBName*

Hinzufügen einer Azure-Cosmos-DB-Datenquelle

Fügen Sie einen Knoten Datenquelle – Azure Cosmos DB wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Azure-Cosmos-DB-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Azure-Cosmos-DB-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Azure-Cosmos-DB-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie Cosmos DB-Datenbankname — geben Sie den Namen der Datenbank an, aus der Sie lesen möchten,. *cosmosDBName*
3. Wählen Sie Azure Cosmos DB Container — geben Sie den Namen des Containers ein, aus dem Sie lesen möchten,. *cosmosContainerName*
4. Wählen Sie optional Benutzerdefinierte Abfrage in Azure Cosmos DB aus – geben Sie eine SQL-SELECT-Abfrage ein, um bestimmte Informationen aus Azure Cosmos DB abzurufen.
5. Geben Sie unter Benutzerdefinierte Azure-Cosmos-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines Azure-Cosmos-DB-Zielknotens

Voraussetzungen

- Eine AWS Glue Azure Cosmos DB-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Azure-Cosmos-DB-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.

- Eine Azure-Cosmos-DB-Tabelle, in die Sie schreiben möchten. Sie benötigen Identifikationsinformationen für den Container. Sie müssen den Container erstellen, bevor Sie die Verbindungsmethode aufrufen.

  Ein Container in Azure Cosmos DB für NoSQL wird anhand seiner Datenbank und seines Containers identifiziert. Sie müssen die Datenbank- und Container-Namen angeben*cosmosContainerName*, wenn Sie eine Verbindung zur Azure Cosmos for NoSQL-API herstellen. *cosmosDBName*

Hinzufügen eines Azure-Cosmos-DB-Datenziels

Fügen Sie einen Knoten Datenziel – Azure Cosmos DB wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Azure-Cosmos-DB-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Azure-Cosmos-DB-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Azure-Cosmos-DB-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie Cosmos DB-Datenbankname — geben Sie den Namen der Datenbank an, aus der Sie lesen möchten,. *cosmosDBName*
3. Wählen Sie Azure Cosmos DB Container — geben Sie den Namen des Containers ein, aus dem Sie lesen möchten,. *cosmosContainerName*
4. Geben Sie unter Benutzerdefinierte Azure-Cosmos-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines Azure-Cosmos-DB-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie die, die bei der Programmierung AWS Glue für Spark-Skripte verfügbar sind.

Siehe the section called “Azure-Cosmos-DB-Verbindungen”.

Verbindung zu Azure SQL herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für Azure SQL. AWS Glue Studio bietet eine visuelle Oberfläche, über die Sie eine Verbindung zu Azure SQL herstellen, Datenintegrationsjobs erstellen und diese auf dem ausführen können AWS Glue Studio serverlose Spark-Laufzeit.

Themen

- Erstellen einer Azure-SQL-Verbindung
- Erstellen eines Azure-SQL-Quellknotens
- Erstellen eines Azure-SQL-Zielknotens
- Erweiterte Optionen

Erstellen einer Azure-SQL-Verbindung

Um von aus eine Verbindung zu Azure SQL herzustellen AWS Glue, müssen Sie Ihre Azure SQL-Anmeldeinformationen erstellen und in einem AWS Secrets Manager geheimen Ordner speichern und dieses Geheimnis dann einer Azure AWS Glue SQL-Verbindung zuordnen.

Eine Verbindung zu Azure SQL konfigurieren Sie wie folgt:

1. Erstellen Sie unter mithilfe Ihrer Azure SQL-Anmeldeinformationen ein Geheimnis. AWS Secrets Manager Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *`azuresqlUsername`*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *`azuresqlPassword`*
2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen,*`connectionName`*, für die future Verwendung in AWS Glue.

   - Wählen Sie als Verbindungstyp Azure SQL aus.
   - Wenn Sie eine Azure-SQL-URL angeben, geben Sie eine JDBC-Endpunkt-URL an.

Die URL muss das folgende Format aufweisen: `jdbc:sqlserver://databaseServerName:databasePort;databaseName=azuresqlDBnam`

AWS Glue erfordert die folgenden URL-Eigenschaften:

- `databaseName` – Eine Standarddatenbank in Azure SQL, mit der eine Verbindung hergestellt werden kann.

Weitere Informationen zu JDBC URLs für Azure SQL Managed Instances finden Sie in der Microsoft-Dokumentation.

- Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an: *secretName*

Erstellen eines Azure-SQL-Quellknotens

Voraussetzungen

- Eine AWS Glue Azure SQL-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Azure-SQL-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine Azure SQL-Tabelle, aus der Sie lesen möchten,*tableName*.

  Eine Azure-SQL-Tabelle wird anhand ihrer Datenbank, ihres Schemas und ihres Tabellennamens identifiziert. Sie müssen den Datenbanknamen und den Tabellennamen angeben, wenn Sie eine Verbindung zu Azure SQL herstellen. Sie müssen auch das Schema angeben, falls es sich nicht um das Standardschema „public“ handelt. Die Datenbank wird über eine URL-Eigenschaft in*connectionName*, ein Schema und einen Tabellennamen über die bereitgestelltdbtable.

Hinzufügen einer Azure-SQL-Datenquelle

Fügen Sie einen Knoten Datenquelle – Azure SQL wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Azure-SQL-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Azure-SQL-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Azure-SQL-Verbindung”.

Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.

2. Wählen Sie eine Option für Azure-SQL-Quelle aus:

   - Eine einzelne Tabelle auswählen – rufen Sie alle Daten aus einer einzigen Tabelle ab.
   - Benutzerdefinierte Abfrage eingeben – rufen Sie einen Datensatz aus mehreren Tabellen ab, basierend auf Ihrer benutzerdefinierten Abfrage.

3. Wenn Sie eine einzelne Tabelle ausgewählt haben, geben Sie ein*tableName*.

   Wenn Sie Benutzerdefinierte Abfrage eingeben ausgewählt haben, geben Sie eine TransactSQL-SELECT-Abfrage ein.

4. Geben Sie unter Benutzerdefinierte Azure-SQL-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines Azure-SQL-Zielknotens

Voraussetzungen

- Eine AWS Glue Azure SQL-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Azure-SQL-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine Azure SQL-Tabelle, in die Sie schreiben möchten,*tableName*.

  Eine Azure-SQL-Tabelle wird anhand ihrer Datenbank, ihres Schemas und ihres Tabellennamens identifiziert. Sie müssen den Datenbanknamen und den Tabellennamen angeben, wenn Sie eine Verbindung zu Azure SQL herstellen. Sie müssen auch das Schema angeben, falls es sich nicht um das Standardschema „public“ handelt. Die Datenbank wird über eine URL-Eigenschaft in*connectionName*, ein Schema und einen Tabellennamen über die bereitgestelltdbtable.

Hinzufügen eines Azure-SQL-Datenziels

Fügen Sie einen Knoten Datenziel – Azure SQL wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Azure-SQL-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Azure-SQL-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Azure-SQL-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Konfigurieren Sie den Tabellennamen, indem Sie angeben`tableName`.
3. Geben Sie unter Benutzerdefinierte Azure-SQL-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines Azure-SQL-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie die, die bei der Programmierung AWS Glue für Spark-Skripte verfügbar sind.

Siehe the section called “Azure-SQL-Verbindungen”.

Verbindung zu Blackbaud Raiser's Edge NXT herstellen

Edge NXT von Blackbaud Raiser ist eine umfassende Cloud-basierte Softwarelösung für Fundraising und Spenderverwaltung, die speziell für gemeinnützige Organisationen und die gesamte gemeinnützige Gemeinschaft entwickelt wurde. Dieser Konnektor basiert auf der SKY-API von Blackbaud Raiser Edge NXT und bietet Funktionen zur Verwaltung von Entitäten, die sich in Raisers Edge NXT befinden.

Themen

- AWS Glue Unterstützung für Blackbaud Raiser's Edge NXT
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Blackbaud Raiser's Edge NXT
- Konfiguration der Edge NXT-Verbindungen von Blackbaud Raiser
- Aus Blackbaud Raisers Edge NXT-Entitäten lesen

- Edge NXT-Verbindungsoptionen von Blackbaud Raiser
- Einschränkungen von Blackbaud Raiser's Edge NXT

AWS Glue Unterstützung für Blackbaud Raiser's Edge NXT

AWS Glue unterstützt Blackbaud Raiser's Edge NXT wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Blackbaud Raisers Edge NXT abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte Edge NXT API-Versionen von Blackbaud Raiser

Die folgenden Edge NXT API-Versionen von Blackbaud Raiser werden unterstützt:

- v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
```

```
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Blackbaud Raiser's Edge NXT

Bevor Sie Daten von Blackbaud Raiser's Edge NXT übertragen können, AWS Glue müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Blackbaud Raiser's Edge NXT-Konto.
- Sie haben in Ihrem Blackbaud Raiser's Edge NXT-Konto ein Zugriffstoken generiert, dem der entsprechende Lese-/Schreibbereich für den Zugriff zugewiesen wurde. APIs Weitere Informationen finden Sie unter Autorisierung.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Blackbaud Raiser's Edge NXT-Konto herstellen.

Konfiguration der Edge NXT-Verbindungen von Blackbaud Raiser

Edge NXT von Blackbaud Raiser unterstützt den Grant-Typ AUTHORIZATION_CODE für. OAuth2

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Blackbaud Raiser's Edge NXT weiter, wo der Benutzer sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Blackbaud Raiser's Edge NXT-Instanz gewähren muss.
- Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Blackbaud Raiser's Edge NXT zu erstellen und beim Herstellen von Verbindungen über die Konsole ihre eigene Client-ID, ihren Abonnementschlüssel und ihre eigene Instanz-URL anzugeben. AWS Glue In diesem Szenario werden sie weiterhin zu Blackbaud Raiser's Edge NXT weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen zu autorisieren. AWS Glue
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Die öffentliche Blackbaud Raiser's Edge NXT-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter Autorisierung.

So konfigurieren Sie eine Blackbaud Raiser's Edge NXT-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret den API-Schlüssel der verbundenen App mit einem *USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET* Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Blackbaud Raiser's Edge NXT aus.
   b. Geben Sie das `INSTANCE_URL` Blackbaud Raiser's Edge NXT-Konto an, mit dem Sie eine Verbindung herstellen möchten.
   c. Stellen Sie die vom Benutzer verwaltete Client-Anwendung bereit. `clientId`

d. Geben Sie den Abonnementschlüssel an, der mit Ihrem Konto verknüpft ist.

e. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

f. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Blackbaud Raisers Edge NXT-Entitäten lesen

Voraussetzung

Ein Blackbaud Raiser's Edge NXT-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Konstituierende Adresse	Ja	Ja	Nein	Ja	Ja
Konstituierende Bildung	Ja	Ja	Nein	Ja	Ja
Konstituierende E-Mail-Adresse	Ja	Ja	Nein	Ja	Ja
Konstituierendes Telefon	Ja	Ja	Nein	Ja	Ja
Konstituierende Anmerkung	Ja	Ja	Nein	Ja	Ja
Konstituierende Beziehung	Ja	Ja	Nein	Ja	Ja
Konstituierende Online-Präsenz	Ja	Ja	Nein	Ja	Ja
Gelegenheit	Ja	Ja	Nein	Ja	Ja
Berufung einlegen	Ja	Ja	Nein	Ja	Ja
Kampagne	Ja	Ja	Nein	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Fonds	Ja	Ja	Nein	Ja	Ja
Paket	Ja	Ja	Nein	Ja	Ja
Geschenkstapel	Ja	Ja	Nein	Ja	Nein
Teilnehmer der Veranstaltung	Ja	Ja	Ja	Ja	Ja
Konstituierende Zuordnung der Spendenaktion	Nein	Nein	Nein	Ja	Nein
Geschenk	Ja	Ja	Ja	Ja	Ja
Mitgliedschaften	Ja	Ja	Nein	Ja	Ja
Aktion	Ja	Ja	Nein	Ja	Nein
Konstituent	Ja	Ja	Ja	Ja	Ja
Konstituierende Güter	Ja	Ja	Nein	Ja	Ja
Ereignis	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Benutzerdefiniertes Feld für Geschenke	Ja	Ja	Nein	Ja	Ja

Beispiel:

```
blackbaud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="BLACKBAUD",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1",
        "SUBSCRIPTION_KEY": <Subscription key associated with one's developer account>
    }
```

Details zur Entität und zum Feld von Blackbaud Raiser's Edge NXT

Weitere Informationen zu den Entitäten und Felddetails finden Sie unter:

- Action (Aktion)
- Konstituent
- Konstituierende Adresse
- Konstituierende Mitgliedschaft
- Konstituierende Zuordnung der Spendenaktion
- Konstituierende Bildung
- Konstituierende E-Mail-Adresse
- Konstituierendes Telefon
- Konstituierende Anmerkung
- Konstituierende Online-Präsenz
- Konstituierende Beziehung
- Veranstaltung

- Teilnehmer der Veranstaltung
- Berufung einlegen
- Kampagne
- Fonds
- Package
- Geschenk
- Benutzerdefiniertes Feld für Geschenke
- Geschenkstapel
- Gelegenheit
- Konstituierende Codes

> **Note**
>
> Die Datentypen Struct und List werden in den Datentyp String konvertiert, und der DateTime Datentyp wird in der Antwort der Konnektoren in Timestamp konvertiert.

Abfragen partitionieren

Feldbasierte Partitionierung:

Edge NXT von Blackbaud Raiser unterstützt keine feldbasierte oder datensatzbasierte Partitionierung.

Datensatzbasierte Partitionierung:

Sie können die zusätzliche Spark-Option bereitstellen, `NUM_PARTITIONS` wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Bei der datensatzbasierten Partitionierung wird die Gesamtzahl der vorhandenen Datensätze von der Edge NXT API von Blackbaud Raiser abgefragt und durch die angegebene Anzahl dividiert. `NUM_PARTITIONS` Die resultierende Anzahl von Datensätzen wird dann gleichzeitig von jeder Unterabfrage abgerufen.

- `NUM_PARTITIONS`: die Anzahl der Partitionen.

Beispiel:

```
blackbaud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="BLACKBAUD",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "2",
        "SUBSCRIPTION_KEY": <Subscription key associated with one's developer account>
    }
```

Edge NXT-Verbindungsoptionen von Blackbaud Raiser

Im Folgenden sind die Verbindungsoptionen für den Edge NXT von Blackbaud Raiser aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Blackbaud Raiser's Edge NXT.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Edge NXT Rest-API-Version von Blackbaud Raiser, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `NUM_PARTITIONS`(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen. Beispielwert: 10.
- `SUBSCRIPTION_KEY`(Zeichenfolge) — (Erforderlich) Standard: leer. Wird zum Lesen verwendet. Abonnementschlüssel, der mit dem eigenen Entwicklerkonto verknüpft ist.

Einschränkungen von Blackbaud Raiser's Edge NXT

Im Folgenden finden Sie Einschränkungen oder Hinweise für Blackbaud Raiser's Edge NXT:

- SaaS unterstützt nur den `EQUAL_TO` Operator, der Ergebnisse zurückgibt, die am oder nach dem angegebenen Datum erstellt oder geändert wurden. Darüber hinaus ist das Feld „id“ ein String-

Datentyp. Es gibt auch keine Identifizierung von Feldern, die keine NULL-Werte zulassen. Daher wird feldbasierte Partitionierung nicht unterstützt.

- Inkrementelles Abrufen wird von der `Event` Entität nur mit täglichen, monatlichen und wöchentlichen Frequenzen unterstützt.
- Die Entität Constituent Fundraiser Assignment gibt maximal 20 Datensätze zurück.
- Datensatzbasierte Partitionierung:
  - Wird von den `Action` Entitäten `Constituent Fundraiser Assignment` oder `Gift Batch` nicht unterstützt.
  - Die datensatzbasierte Partitionierung mit dem Filterprädikat wird nur von den `Event` Entitäten und unterstützt. `Event Participant` Wenn ein Filterprädikat mit anderen datensatzbasierten unterstützten Entitäten verwendet wird, wird eine Ausnahme ausgelöst.
- In der `Gift Custom Field` Entität muss das Feld 'Wert' in Verbindung mit dem Feld 'Kategorie' verwendet werden, was andernfalls zu einer ungefilterten Antwort führt. Um also den Benutzer zu zwingen, das Feld „Kategorie“ einzugeben und gleichzeitig mit dem Feld „Wert“ zu filtern, wird eine Ausnahme ausgelöst, wenn die oben genannte Anforderung nicht erfüllt wurde.
- Die `last_modified` Felder `date_added` und für alle zutreffenden Entitäten unterstützen keine Vergleichsoperatoren. Sie unterstützen nur den Operator „Equal to“. Außerdem gibt es kein Feld, das mit den oben genannten Feldern kombiniert werden kann, um eine Reihe von Datensätzen bereitzustellen. Daher sind diese Felder nur abfragbar und unterstützen keine inkrementelle Übertragung.
- Das `added_by` Feld in der `Gift Batch` Entität wird nicht als filterbar betrachtet, da es möglicherweise nicht die richtigen Ergebnisse ausgibt.
- Beim Einfügen von Daten in die `Gift` Entität besteht eine Latenz von etwa 30 Minuten für das Abrufen von Datensätzen über den `/GET Gift List` Endpunkt.
- Die Support für die inkrementelle Übertragung wurde für die Gift-Entität aufgrund von Einschränkungen seitens der Datenquelle eingestellt.
- Für das Statusfeld in der Opportunity-Entität besteht eine Latenz von 10 Minuten.
- Die `Fundraiser Assignment` Entität hat `Constituent` als abhängige Entität. Der Konnektor lädt maximal 5.000, aus IDs denen ausgewählt werden kann, um zu verhindern, dass die Antwortgröße die maximal zulässige Nutzdatengröße überschreitet.

Verbindung zu CircleCI herstellen

CircleCI ist eine Plattform für kontinuierliche Integration und kontinuierliche Bereitstellung. Ihr CircleCI-Konto enthält Daten zu Ihren Projekten, Pipelines, Workflows und mehr. Wenn Sie ein CircleCI-Benutzer sind, können Sie sich mit Ihrem CircleCI-Konto AWS Glue verbinden. Anschließend können Sie CircleCI als Datenquelle in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen CircleCI und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für CircleCI
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- CircleCI konfigurieren
- CircleCI-Verbindungen konfigurieren
- Aus CircleCI-Entitäten lesen
- CircleCI-Verbindungsoptionen
- CircleCI-Einschränkungen

AWS Glue Unterstützung für CircleCI

AWS Glue unterstützt CircleCI wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von CircleCI abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte CircleCI API-Versionen

Die folgenden CircleCI-API-Versionen werden unterstützt:

- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
     ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben

Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

CircleCI konfigurieren

Bevor Sie Daten von CircleCI AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Konto bei CircleCI, das die Daten enthält, die Sie übertragen möchten.
- In den Benutzereinstellungen für Ihr Konto haben Sie ein persönliches API-Token erstellt. Weitere Informationen finden Sie unter Ein persönliches API-Token erstellen.
- Sie geben das persönliche API-Token an, AWS Glue während Sie die Verbindung herstellen.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem CircleCI-Konto herstellen.

CircleCI-Verbindungen konfigurieren

CircleCI unterstützt benutzerdefinierte Authentifizierung.

Um eine CircleCI-Verbindung zu konfigurieren:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret den API-Schlüssel der verbundenen App mit einem `Circle-Token` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle CircleCI aus.
   b. Wählen Sie die AWS IAM-Rolle aus, die die AWS Glue folgenden Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
```

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

c. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

d. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus CircleCI-Entitäten lesen

Voraussetzung

Ein CircleCI-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kontext	Ja	Nein	Nein	Ja	Nein
Metrik zur Zusammenfassung der Organisation	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Pipeline	Nein	Nein	Nein	Ja	Nein
Pipeline-Workflow	Ja	Nein	Nein	Ja	Nein
Zweig des Projekts	Ja	Nein	Nein	Ja	Nein
Projekt Flaky Test	Nein	Nein	Nein	Ja	Nein
Metrik zur Projektzusammenfassung	Ja	Nein	Nein	Ja	Nein
Plan	Nein	Nein	Nein	Ja	Nein
Workflow-Job-Zeitreihen	Ja	Nein	Nein	Ja	Nein
Workflow-Metrik und Trend	Ja	Nein	Nein	Ja	Nein
Kürzlich ausgeführter Workflow	Ja	Nein	Nein	Ja	Nein
Metrik zur Workflow-Zusammenfassung	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Metrik für den Workflow-Test	Ja	Nein	Nein	Ja	Nein

Beispiel:

```
circleci_read = glueContext.create_dynamic_frame.from_options(
    connection_type="circleci",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "context/e7ea2945-dccb-4205-b673-8391fe1b3a4c",
        "API_VERSION": "v2"
    }
```

CircleCI-Entitäts- und Felddetails

Weitere Informationen zu den Entitäten und Felddetails finden Sie unter:

- Kontexte
- Kennzahlen zur Projektzusammenfassung
- Workflow-Job-Zeitreihen
- Kennzahlen im Überblick über die Organisation
- Zweigstellen des Projekts
- Projekt Flaky Tests
- Arbeitsablauf Letzte Läufe
- Metriken zur Workflow-Zusammenfassung
- Workflow-Metriken und Trends
- Metriken für Workflow-Tests
- Pipelines
- Pipeline-Arbeitsabläufe
- Zeitpläne

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontext	Erstellt am	String	
	ID	String	
	Name	String	
	Typ des Besitzers	String	EQUAL_TO
Metrik zur Zusammenfassung der Organisation	Alle Projekte	Auflisten	
	Daten der Organisation	Struct	
	Projektdaten der Organisation	Auflisten	
	Namen der Projekte	String	EQUAL_TO
	Berichtsfenster	String	EQUAL_TO
Pipeline	Verzweigung	String	EQUAL_TO
	Erstellt am	String	
	Fehler	Auflisten	
	ID	String	
	Zahl	Ganzzahl	
	Projekt Slug	String	
	Status	String	
	Auslöser	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Trigger-Parameter	Struct	
	Aktualisiert am	String	
	VCS	Struct	
Pipeline-Workflow	Storniert von	String	
	Erstellt am	String	
	Fehler von	String	
	ID	String	
	Name	String	
	Pipeline-ID	String	
	Pipeline-Nummer	Ganzzahl	
	Projekt Slug	String	
	Gestartet von	String	
	Status	String	
	Gestoppt bei	String	
	Markierung	String	
Zweigstelle des Projekts	Branches	Auflisten	
	ID der Organisation	String	
	Projekt-ID	String	
	Workflow-Name	String	EQUAL_TO
Projekt Flaky Test	Klassenname	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Datei	String	
	Auftragsname	String	
	Nummer des Auftrags	Ganzzahl	
	Pipeline-Nummer	Ganzzahl	
	Quelle	String	
	Name des Tests	String	
	Verschwendete Zeit	Ganzzahl	
	Die Zeiten sind abgeflaut	Ganzzahl	
	Der Workflow wurde erstellt am	String	
	Workflow-ID	String	
	Workflow-Name	String	
Metrik zur Projektzusammenfassung	Alle Branchen	Auflisten	
	Alle Workflows	Auflisten	
	Branches	String	EQUAL_TO
	ID der Organisation	String	
	Daten des Projekts	Struct	
	Projekt-ID	String	
	Daten der Zweigstelle des Projekt-Workflows	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Daten zum Projekt-Workflow	Auflisten	
	Berichtsfenster	String	EQUAL_TO
	Workflow-Namen	String	EQUAL_TO
Plan	Akteur	Struct	
	Erstellt am	String	
	Beschreibung	String	
	ID	String	
	Name	String	
	Parameter	Struct	
	Projekt Slug	String	
	Zeitplan	Struct	
	Aktualisiert am	String	
Workflow-Job-Zeitreihen	Verzweigung	String	EQUAL_TO
	Granularity	String	EQUAL_TO
	Max endete am	String	
	Metriken	Struct	
	Min begann am	String	
	Name	String	
	Start- und Enddatum	DateTime	EQUAL_TO, ZWISCHEN

Entität	Feld	Datentyp	Unterstützte Operatoren
	Zeitstempel	String	
Workflow-Metrik und Trend	Alle Branchen	Boolesch	EQUAL_TO
	Branches	String	EQUAL_TO
	Metriken	Struct	
	Tendenzen	Struct	
	Workflow-Namen	Auflisten	
Kürzlich ausgeführte Workflow-Ausführung	Alle Filialen	Boolesch	EQUAL_TO
	Verzweigung	String	EQUAL_TO
	Erstellt am	String	
	Verwendete Credits	Ganzzahl	
	Dauer	Ganzzahl	
	ID	String	
	Ist Genehmigung	Boolesch	
	Start- und Enddatum	DateTime	EQUAL_TO, ZWISCHEN
	Status	String	
	Gestoppt bei	String	
Metrik zur Workflow-Zusammenfassung	Alle Branchen	Boolesch	EQUAL_TO
	Verzweigung	String	EQUAL_TO
	Metriken	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Name	String	
	Projekt-ID	String	
	Berichtsfenster	String	EQUAL_TO
	Ende des Fensters	String	
	Anfang des Fensters	String	
Metrik zum Workflow-Test	Durchschnittliche Anzahl der Tests	Ganzzahl	
	Verzweigung	String	EQUAL_TO
	Die meisten fehlgeschlagenen Tests	Auflisten	
	Die meisten fehlgeschlagenen Tests Extra	Ganzzahl	
	Langsamste Tests	Auflisten	
	Langsamste Tests Extra	Ganzzahl	
	Testläufe	Auflisten	
	Testläufe insgesamt	Ganzzahl	

> **Note**
>
> Die Datentypen Struct und List werden in der Antwort des Konnektors in den Datentyp String konvertiert.

Abfragen partitionieren

CircleCI unterstützt keine feldbasierte oder datensatzbasierte Partitionierung.

CircleCI-Verbindungsoptionen

Im Folgenden sind Verbindungsoptionen für CircleCI aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in CircleCI.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. CircleCI Rest API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

CircleCI-Einschränkungen

Im Folgenden finden Sie Einschränkungen oder Hinweise für CircleCI:

- CircleCI unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.
- Filterfelder, die '-' (Bindestrich) enthalten, funktionieren nur, wenn sie in Backticks eingeschlossen sind. Zum Beispiel: `workflow-name` = „abc“
- Der GitLab VCS-Typ kann nicht unterstützt werden, da es keine programmatische Möglichkeit gibt, die für den VCS-Entitätspfad erforderliche 'Projekt-ID' abzurufen. GitLab

Verbindung zu Datadog herstellen

Datadog ist eine Überwachungs- und Analyseplattform für Cloud-Anwendungen, einschließlich Infrastruktur, Anwendungen, Dienste und Tools.

Themen

- AWS Glue Unterstützung für Datadog
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Datadog konfigurieren

- Konfiguration von Datadog-Verbindungen
- Aus Datadog-Entitäten lesen
- Datadog-Verbindungsoptionen
- Ein Datadog-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Datadog

AWS Glue unterstützt Datadog wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Datadog abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Datadog API-Versionen

- v1
- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Datadog konfigurieren

Bevor Sie die Übertragung von Datadog verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Datadog-Konto mit und. DD-API-KEY DD-APPLICATION-KEY Weitere Informationen zum Erstellen eines Kontos finden Sie unter Datadog-Konto erstellen.
- Ihr Datadog-Konto hat API-Zugriff mit gültiger Lizenz.

Datadog unterstützt die folgenden sechs. URLs Alle Datadog-API-Clients sind standardmäßig so konfiguriert, dass sie die Datadog-Site nutzen. US1 APIs Wenn Sie sich auf der Datadog EU-Site befinden, müssen Sie die URL https://api.datadoghq.eu mit dem Ende der `DD-API-KEY` Datadog EU-Site auswählen, um auf die zuzugreifen. `DD-APPLICATION-KEY` APIs In ähnlicher Weise sollten Sie für andere Websites die jeweilige URLs mit `DD-API-KEY and DD-APPLICATION-KEY` der jeweiligen Site auswählen.

- US1 API-URL — https://api.datadoghq.com https://api.datadoghq.com
- API-URL der EU — https://api.datadoghq.eu
- US3 API-URL — https://api.us3.datadoghq.com
- US5 API-URL — https://api.us5.datadoghq.com
- API-URL MIT S1-FED — https://api.ddog-gov.com
- Japan API-URL — https://api.ap1.datadoghq.com

Wenn Sie diese Anforderungen erfüllen, sind Sie bereit, eine Verbindung AWS Glue zu Ihrem Datadog-Konto herzustellen.

Konfiguration von Datadog-Verbindungen

Datadog unterstützt die benutzerdefinierte Authentifizierung. Im Folgenden finden Sie die Schritte zur Konfiguration der Datadog-Verbindung:

So konfigurieren Sie eine Datadog-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:

   Für vom Kunden verwaltete verbundene Apps — Secret sollte die verbundene App Consumer Secret mit **`API_KEY`** und **`APPLICATION_KEY`** als Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, pro Verbindung in ein Geheimnis zu erstellen AWS Glue.

2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Datadog aus.
   b. Geben Sie den `Instance_Url` Datadog an, zu dem Sie eine Verbindung herstellen möchten.

c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Datadog-Entitäten lesen

Voraussetzungen

Ein Datadog-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Zeitreihen von Metriken	Ja	Nein	Nein	Ja	Nein
Abfragen protokollieren	Ja	Ja	Ja	Ja	Nein

Beispiel

```
Datadog_read = glueContext.create_dynamic_frame.from_options(
    connection_type="datadog",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "log-queries",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://api.datadoghq.com",
        "FILTER_PREDICATE": "from = `2023-10-03T09:00:26Z`"
    }
```

Datadog-Entitäts- und Felddetails

Entität	Feld	Datentyp	Unterstützte Betreiber
Zeitreihen mit Metriken	error	String	N/A
	aggr	String	N/A
	Attribute	Struct	N/A
	Anzeige_name	String	N/A
	end	DateTime	N/A
	Ausdruck	String	N/A
	Intervall	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Betreiber
	length	Ganzzahl	N/A
	Metrik	String	N/A
	Punktliste	Auflisten	N/A
	Abfrageindex	Ganzzahl	N/A
	scope	String	N/A
	start	DateTime	N/A
	tag_set	Auflisten	N/A
	Einheit	Struct	N/A
	von_bis_datum	DateTime	BETWEEN
	query	String	EQUAL_TO
	Status	String	N/A
	Typ	String	N/A
	Host	String	N/A
Abfragen protokollieren	id	String	N/A
	Attribute	Struct	N/A
	Zeitstempel	DateTime	N/A
	Typ	String	N/A
	from	DateTime	ZWISCHEN, EQUAL_TO
	Indizes	Auflisten	EQUAL_TO
	Speicherebene	String	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Betreiber
	query	String	EQUAL_TO

Datadog-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Datadog aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Datadog.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Datadog Rest API-Version, die Sie verwenden möchten. `v1`Version unterstützt `metrics-timeseries` Entität, während `v2` Version Entität unterstützt`log-queries`.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. URL der Datadog-Instanz. Die URL der Datadog-Instanz variiert je nach Region.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Ein Datadog-Konto erstellen

1. Gehe zu/. https://www.datadoghq.com
2. Wählen Sie JETZT KOSTENLOS LOSLEGEN.
3. Geben Sie die erforderlichen Informationen ein und melden Sie sich an.
4. Installieren Sie den Datadog Agent Installer wie vorgeschlagen.
5. Stellen Sie sicher, dass das Konto bei einer gültigen Organisation (aus der verfügbaren Liste) registriert wird, die über ein aktives Datadog-Abonnement verfügt.
6. Nachdem Sie sich bei Ihrem Datadog-Konto angemeldet haben, bewegen Sie den Mauszeiger über Ihren Benutzernamen in der oberen rechten Ecke, um die Schlüsseldetails anzuzeigen:

   a. Um Ihren API-Schlüssel zu erhalten, wählen Sie API-Schlüssel.

b. Um Ihren Anwendungsschlüssel zu erhalten, wählen Sie Anwendungsschlüssel.

Einschränkungen

Die folgenden Einschränkungen gelten für den Datadog-Konnektor:

- Datadog unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.
- `from`ist ein obligatorischer Filterparameter für die Entität. `Log Queries`
- `from_to_date`und `query` sind obligatorische Filterparameter für die `Metrics Timeseries` Entität.

Verbindung zu Docusign Monitor herstellen

Docusign Monitor hilft Unternehmen dabei, ihre Vereinbarungen durch Aktivitätsverfolgung zu schützen. round-the-clock Die Monitor-API stellt diese Informationen zur Aktivitätsverfolgung direkt an bestehende Sicherheitsstapel oder Datenvisualisierungstools bereit. So können Teams unbefugte Aktivitäten erkennen, Vorfälle untersuchen und schnell auf verifizierte Bedrohungen reagieren. Sie bietet außerdem die Flexibilität, die Sicherheitsteams benötigen, um Dashboards und Warnmeldungen an spezifische Geschäftsanforderungen anzupassen.

Themen

- AWS Glue Unterstützung für Docusign Monitor
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Docusign Monitor konfigurieren
- Docusign Monitor-Verbindungen konfigurieren
- Aus Docusign Monitor-Entitäten lesen
- Verbindungsoptionen für Docusign Monitor
- Einschränkungen von Docusign Monitor

AWS Glue Unterstützung für Docusign Monitor

AWS Glue unterstützt Docusign Monitor wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Docusign Monitor abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Docusign Monitor-API-Versionen

Die folgenden Docusign Monitor-API-Versionen werden unterstützt:

- v2.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Docusign Monitor konfigurieren

Bevor Sie Daten von Docusign Monitor AWS Glue an unterstützte Ziele übertragen können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Docusign-Konto, mit dem Sie das Docusign-Softwareprodukt in Docusign Monitor verwenden.
- In der Entwicklerkonsole für Ihr Docusign-Konto haben Sie eine 2.0-Integrations-App für erstellt. OAuth AWS Glue

  Diese App stellt die Kundenanmeldedaten bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter OAuth 2.0 in der Docusign Monitor-Dokumentation.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Docusign Monitor-Konto herstellen.

Docusign Monitor-Verbindungen konfigurieren

Docusign Monitor unterstützt den Zuschusstyp AUTHORIZATION_CODE.

- Dieser Gewährungstyp wird als dreigeteilt betrachtet, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Docusign Monitor zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Docusign Monitor weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen zu autorisieren AWS Glue .
- Dieser Grant-Typ führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Die öffentliche Dokumentation von Docusign Monitor zur Erstellung einer verbundenen App für den OAuth Autorisierungscode-Flow finden Sie unter Docusign OAuth Connect.

So konfigurieren Sie eine Docusign Monitor-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret den API-Schlüssel der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Verbindungen die Option Verbindung erstellen.
   b. Wählen Sie bei der Auswahl einer Datenquelle Docusign Monitor aus.
   c. Wählen Sie die AWS IAM-Rolle aus, die die AWS Glue folgenden Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Stellen Sie die benutzerverwaltete Client-Anwendung ClientId der Docusign Monitor-App bereit.

e. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

f. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Docusign Monitor-Entitäten lesen

Voraussetzung

Ein Docusign Monitor-Objekt, aus dem Sie lesen möchten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Überwachen von Daten	Ja	Ja	Nein	Ja	Nein

Beispiel:

```
docusignmonitor_read = glueContext.create_dynamic_frame.from_options(
    connection_type="docusign_monitor",
    connection_options={
        "connectionName": "connectionName",
```

```
        "ENTITY_NAME": "monitoring-data",
        "API_VERSION": "v2.0"
    }
```

Entitäts- und Felddetails von Docusign Monitor

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Überwachen von Daten	Zeitstempel	DateTime	N/A
	eventId	String	N/A
	Anwendung	String	N/A
	Umgebung	String	N/A
	site	String	N/A
	traceToken	String	N/A
	Organisations-ID	String	N/A
	accountId	String	N/A
	userId	String	N/A
	object	String	N/A
	action	String	N/A
	property	String	N/A
	field	String	N/A
	Ergebnis	String	N/A
	IntegratorKey	String	N/A
	Kunde sichtbar	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	version	String	N/A
	userAgent	String	N/A
	userAgentClientInformationen	Struct	N/A
	ipAddress	String	N/A
	ipAddressLocation	Struct	N/A
	data	String	N/A
	Quelle	String	N/A
	Breitengrand	Double	N/A
	Längengrad	Double	N/A
	city	String	N/A
	state	String	N/A
	country	String	N/A
	usUserMemberOfDomain	Boolesch	N/A
	affectedUserIsMemberOfDomain	Boolesch	N/A
	Proxy-Status	String	N/A
	Proxy-Typ	String	N/A
	Proxy-Ebene	String	N/A
	referencedUserId	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Gerät	String	N/A
	Browser	String	N/A
	cursor	DateTime	EQUAL_TO

Abfragen partitionieren

Docusign Monitor unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.

Verbindungsoptionen für Docusign Monitor

Im Folgenden sind die Verbindungsoptionen für Docusign Monitor aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Docusign Monitor.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Docusign Monitor-Rest-API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.

Einschränkungen von Docusign Monitor

Im Folgenden finden Sie Einschränkungen oder Hinweise für Docusign Monitor:

- Wenn mithilfe des `cursor` Felds ein Filter angewendet wird, ruft die API Datensätze für die nächsten sieben Tage ab dem angegebenen Datum ab.
- Wenn kein Filter bereitgestellt wird, ruft die API Datensätze für die letzten sieben Tage ab dem aktuellen Datum der API-Anfrage ab.

- Docusign Monitor unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.
- Docusign Monitor unterstützt die Funktion „Sortieren nach“ nicht.

Verbindung zu Domo herstellen

Domo ist ein Cloud-basiertes Dashboarding-Tool. Mit der Unternehmensanwendungsplattform von Domo ist die Grundlage geschaffen, die Sie für die Erweiterung von Domo benötigen, sodass Sie Ihre individuellen Lösungen schneller erstellen können.

Themen

- AWS Glue Unterstützung für Domo
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Domo konfigurieren
- Konfiguration von Domo-Verbindungen
- Aus Domo-Entitäten lesen
- Domo-Verbindungsoptionen
- Domo-Einschränkungen

AWS Glue Unterstützung für Domo

AWS Glue unterstützt Domo wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Domo abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Domo-API-Versionen

Die folgenden Domo-API-Versionen werden unterstützt:

- v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
     ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben

Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Domo konfigurieren

Bevor Sie Daten von Domo AWS Glue zu unterstützten Zielen übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Domo-Konto, das für den API-Zugriff aktiviert ist.
- Sie haben eine App unter Ihrem Domo-Entwicklerkonto, die die Kundenanmeldedaten bereitstellt, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Aufrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Eine Domo-Entwickler-App erstellen.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Domo-Konto herstellen.

Eine Domo-Entwickler-App erstellen

Um die Client ID und den Client Secret zu erhalten, erstellen Sie ein Entwicklerkonto.

1. Gehen Sie zur Domo-Anmeldeseite für Entwickler.
2. Wählen Sie Login (Anmelden) aus.
3. Geben Sie den Domainnamen ein und klicken Sie auf Weiter.
4. Bewegen Sie den Mauszeiger auf Mein Konto und wählen Sie Neuer Kunde.
5. Geben Sie den Namen und die Beschreibung ein, wählen Sie den Bereich („Daten“) aus und klicken Sie auf Erstellen.
6. Rufen Sie die generierte Client-ID und das Client-Geheimnis von dem neu erstellten Client ab.

Konfiguration von Domo-Verbindungen

Domo unterstützt den Gewährungstyp CLIENT_CREDENTIALS für. OAuth2

- Dieser Gewährungstyp wird als zweigliedrig betrachtet, OAuth da sich nur die Client-Anwendung beim Server authentifiziert, ohne dass der Benutzer involviert ist.
- Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Domo zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue
- Eine öffentliche Domo-Dokumentation zur Erstellung einer verbundenen App für den OAuth Autorisierungscode-Flow finden Sie unter OAuth Authentifizierung.

So konfigurieren Sie eine Domo-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Zugriffstoken für die verbundene App`client_id`, und enthalten`client_secret`.
   b. Hinweis: Sie müssen einen geheimen Schlüssel für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Verbindungen die Option Verbindung erstellen.
   b. Wählen Sie bei der Auswahl einer Datenquelle Domo aus.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Domo-Entitäten lesen

Voraussetzung

Ein Domo-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Datensatz oder Datenberechtigungsrichtlinien. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Dataset	Ja	Ja	Ja	Ja	Ja
Richtlinien für Datenberechtigungen	Nein	Nein	Nein	Ja	Nein

Beispiel:

```
Domo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1"
    }
```

Details zur Domo-Entität und zum Feld

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Richtlinien für Datenberechtigungen	id	Long	N/A
	Typ	String	N/A
	Name	String	N/A
	-Filter	Auflisten	N/A
	Benutzer	Auflisten	N/A
	Virtuelle Benutzer	Auflisten	N/A
	Gruppen	Auflisten	N/A

Für die folgende Entität stellt Domo Endpunkte zum dynamischen Abrufen von Metadaten bereit, sodass die Operatorunterstützung auf der Datentypebene für die Entität erfasst wird.

Entität	Datentyp	Unterstützte Operatoren
Dataset	Ganzzahl	=, !=, <, >, >=, <=
	Long	=, !=, <, >, >=, <=
	String	=,! =, ENTHÄLT
	Datum	=, >, >=, <, <=, ZWISCHEN
	DateTime	=, >, >=, <, <=, ZWISCHEN
	Boolesch	=, !=
	Double	=, !=, <, >, >=, <=

Entität	Datentyp	Unterstützte Operatoren
	Auflisten	N/A
	Struct	N/A

Abfragen partitionieren

Feldbasierte Partitionierung

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2023-01-15T11:18:39.205Z"
  ```

  Für das Datumsfeld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2023-01-15"
  ```
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.

  Beispiel für einen gültigen Wert:

  ```
  "2023-02-15T11:18:39.205Z"
  ```
- NUM_PARTITIONS: die Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt:

Entity-Name	Partitionierung von Feldern	Datentyp
Datensatz	Beliebiges auf Datum/Uhrzeit basierendes Feld [dynamische Metadaten]	DateTime
	Beliebiges datumsbasiertes Feld [dynamische Metadaten]	Datum

Beispiel:

```
Domo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "permissionTime"
        "LOWER_BOUND": "2023-01-15T11:18:39.205Z"
        "UPPER_BOUND": "2023-02-15T11:18:39.205Z"
        "NUM_PARTITIONS": "2"
    }
```

Datensatzbasierte Partitionierung

Sie können die zusätzliche Spark-Option bereitstellen, `NUM_PARTITIONS` wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Bei der datensatzbasierten Partitionierung wird die Gesamtzahl der vorhandenen Datensätze von Domo abgefragt und durch die angegebene Anzahl geteilt. `NUM_PARTITIONS` Die resultierende Anzahl von Datensätzen wird dann gleichzeitig von jeder Unterabfrage abgerufen.

Beispiel:

```
Domo_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "2"
    }
```

Domo-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Domo aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Domo.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. Domo Rest API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Domo-Einschränkungen

Im Folgenden finden Sie Einschränkungen oder Hinweise für Domo:

- Aufgrund einer SDK-Beschränkung funktioniert die Filterung für abfragbare Felder, die mit '_' beginnen (zum Beispiel: _BATCH_ID), nicht wie erwartet.

- Aufgrund einer API-Beschränkung funktioniert die Filterung an dem Datum, das vor dem von Ihnen angegebenen Datum liegt. Dies wirkt sich auch auf den inkrementellen Abruf aus. Um diese Einschränkung zu umgehen, wählen Sie ein Datum, das Ihrer Zeitzone im Vergleich zu UTC entspricht, um Daten für das gewünschte Datum abzurufen.

Verbindung zu Dynatrace herstellen

Dynatrace ist eine Plattform, die Analysen und Automatisierung für umfassende Beobachtbarkeit und Sicherheit bietet. Es ist auf die Überwachung und Optimierung der Anwendungsleistung, der Infrastruktur und der Benutzererfahrung spezialisiert.

Themen

- AWS Glue Unterstützung für Dynatrace
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Dynatrace konfigurieren
- Konfiguration von Dynatrace-Verbindungen
- Aus Dynatrace-Entitäten lesen
- Dynatrace-Verbindungsoptionen
- Einschränkungen von Dynatrace

AWS Glue Unterstützung für Dynatrace

AWS Glue unterstützt Dynatrace wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Dynatrace abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Dynatrace API-Versionen

Die folgenden Dynatrace API-Versionen werden unterstützt:

- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
     ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben

Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Dynatrace konfigurieren

Bevor Sie Daten von Dynatrace AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Dynatrace-Konto.
- Sie haben in Ihrem Dynatrace-Konto ein Zugriffstoken generiert, dem der entsprechende Lese-/Schreibbereich für den Zugriff zugewiesen wurde. APIs Weitere Informationen finden Sie unter Generieren eines Tokens.

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem AWS Glue Dynatrace-Konto verbinden.

Konfiguration von Dynatrace-Verbindungen

Dynatrace unterstützt benutzerdefinierte Authentifizierung.

Um eine Dynatrace-Verbindung zu konfigurieren:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret den API-Schlüssel der verbundenen App mit einem *apiToken* Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Dynatrace aus.
   b. Geben Sie das `INSTANCE_URL` Dynatrace-Konto an, mit dem Sie sich verbinden möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die AWS Glue folgenden Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Dynatrace-Entitäten lesen

Voraussetzung

Ein Dynatrace-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie „Problem“.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Problem	Ja	Ja	Ja	Ja	Nein

Beispiel:

```
Dynatrace_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Dynatrace",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "problem",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://[instanceName].live.dynatrace.com"
    }
```

Entitäts- und Felddetails von Dynatrace:

Dynatrace bietet Endpunkte zum dynamischen Abrufen von Metadaten für unterstützte Entitäten. Dementsprechend wird die Operatorunterstützung auf der Datentypebene erfasst.

Entität	Feld	Datentyp	Unterstützte Operatoren
Problem	Betroffene Entitäten	Auflisten	EQUAL_TO
	ID anzeigen	String	EQUAL_TO
	endTime	DateTime	
	Entitäts-Tags	Auflisten	
	Einzelheiten zu den Beweisen	Struct	
	Analyse der Auswirkungen	Struct	
	Auswirkungsgrad	String	EQUAL_TO
	Betroffene Entitäten	Auflisten	EQUAL_TO
	linkedProblemInfo	Struct	
	Verwaltungszonen	Auflisten	EQUAL_TO
	Problemfilter	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Letzte Kommentare	Struct	
	rootCauseEntity	Struct	EQUAL_TO
	Problem-ID	String	EQUAL_TO
	Schweregrad	String	EQUAL_TO
	startTime	DateTime	BETWEEN
	Status	String	EQUAL_TO
	Titel	String	
	from	DateTime	EQUAL_TO, ZWISCHEN
	problemFilterIds	String	EQUAL_TO
	problemFilterNames	String	EQUAL_TO
	managementZoneIds	String	EQUAL_TO
	text	String	EQUAL_TO
	Wird gewartet	Boolesch	EQUAL_TO
	Nachricht	String	

Abfragen partitionieren

Dynatrace unterstützt keine feld- oder datensatzbasierte Partitionierung.

Dynatrace-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Dynatrace aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Dynatrace.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. Dynatrace Rest API-Version, die Sie verwenden möchten.
- `INSTANCE_URL`(Zeichenfolge) — Wird zum Lesen verwendet. Eine gültige Dynatrace-Instanz-URL.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Einschränkungen von Dynatrace

Im Folgenden finden Sie Einschränkungen oder Hinweise für Dynatrace:

- Dynatrace unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.
- Wenn Sie bei der Funktion „Alles auswählen“ das „Feld“ im Filter angeben, ist es nicht erlaubt, mehr als 10 Datensätze pro Seite zu speichern.
- Die maximal unterstützte Seitengröße beträgt 500. Wenn Sie bei der Erstellung des `evidenceDetails, impactAnalysis, recentComments` Schemas eines der Felder [] auswählen, werden Datensätze pro Seite standardmäßig auf 10 gesetzt.

Verbindung zu Facebook-Werbeanzeigen herstellen

Facebook Ads ist eine leistungsstarke digitale Werbeplattform, die von Unternehmen jeder Größe genutzt wird, um ihre Zielgruppe zu erreichen und verschiedene Marketingziele zu erreichen. Die Plattform ermöglicht es Werbetreibenden, maßgeschneiderte Anzeigen zu erstellen, die in der gesamten App- und Dienstfamilie von Facebook, einschließlich Facebook und Messenger, angezeigt werden können. Mit seinen fortschrittlichen Targeting-Funktionen ermöglicht Facebook Ads Unternehmen, bestimmte demografische Merkmale, Interessen, Verhaltensweisen und Standorte zu erreichen.

Themen

- AWS Glue Unterstützung für Facebook-Werbung
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Facebook-Werbung konfigurieren
- Konfiguration von Facebook Ads-Verbindungen
- Lesen von Facebook Ads-Entitäten
- Verbindungsoptionen für Facebook-Anzeigen
- Einschränkungen und Hinweise für den Facebook Ads Connector

AWS Glue Unterstützung für Facebook-Werbung

AWS Glue unterstützt Facebook-Werbung wie folgt:

Als Quelle unterstützt?

Ja. Du kannst AWS Glue ETL-Jobs verwenden, um Daten aus Facebook-Anzeigen abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen für Facebook-Werbeanzeigen

Die folgenden API-Versionen für Facebook-Werbeanzeigen werden unterstützt:

- v17.0
- v18.0
- v19.0
- v20.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Facebook-Werbung konfigurieren

Bevor du Daten von Facebook-Werbeanzeigen übertragen kannst, musst du die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Auf Facebook-Standardkonten wird direkt über Facebook zugegriffen.
- Zur Generierung des Zugriffstokens ist eine Benutzerauthentifizierung erforderlich.
- Der Facebook Ads SDK-Connector wird den OAuth Ablauf des Benutzerzugriffstokens implementieren.
- Wir verwenden OAuth2 2.0, um unsere API-Anfragen für Facebook-Anzeigen zu authentifizieren. Diese webbasierte Authentifizierung fällt unter die Multi-Factor Authentication (MFA) -Architektur, die eine Obergruppe von 2FA darstellt.
- Der Benutzer muss Berechtigungen für den Zugriff auf die Endpunkte gewähren. Für den Zugriff auf die Benutzerdaten erfolgt die Autorisierung der Endgeräte über Berechtigungen und Funktionen.

Anmeldeinformationen für OAuth Version 2.0 abrufen

Informationen zum Abrufen von API-Anmeldeinformationen, mit denen du authentifizierte Aufrufe an deine Instance tätigen kannst, findest du unter REST-API im Facebook Ads Developer Guide.

Konfiguration von Facebook Ads-Verbindungen

Facebook Ads unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2

- Diese Art der Gewährung wird als dreigeteilt betrachtet, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Facebook Ads zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Facebook Ads weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Diese Art der Gewährung führt zu einem Zugriffstoken. Ein ablaufendes Systembenutzertoken ist ab einem generierten oder aktualisierten Datum 60 Tage lang gültig. Um Kontinuität zu gewährleisten, sollte der Entwickler das Zugriffstoken innerhalb von 60 Tagen aktualisieren. Andernfalls verfällt das Zugriffstoken und der Entwickler muss ein neues erwerben, um den API-Zugriff wiederherzustellen. Siehe Zugriffstoken aktualisieren.
- Die öffentliche Dokumentation von Facebook Ads zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow findest du unter Using OAuth 2.0 to Access Google APIs im Leitfaden Google für Entwickler.

So konfigurierst du eine Facebook Ads-Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Facebook-Werbung aus.
   b. Geben Sie `INSTANCE_URL` die Facebook-Werbeinstanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen von Facebook Ads-Entitäten

Voraussetzung

Ein Facebook-Werbeobjekt, aus dem du lesen möchtest. Sie benötigen den Objektnamen. Die folgenden Tabellen zeigen die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kampagne	Ja	Ja	Nein	Ja	Ja
Anzeigenset	Ja	Ja	Nein	Ja	Ja
Werbeanzeigen	Ja	Ja	Nein	Ja	Ja
Und kreativ	Nein	Ja	Nein	Ja	Nein
Insights — Konto	Nein	Ja	Nein	Ja	Nein
Konten hinzufügen	Ja	Ja	Nein	Ja	Nein
Einblicke - Anzeige	Ja	Ja	Nein	Ja	Ja
Einblicke - AdSet	Ja	Ja	Nein	Ja	Ja
Einblicke — Kampagne	Ja	Ja	Nein	Ja	Ja

Beispiel:

```
FacebookAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="FacebookAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0"
    }
```

Entitäts- und Felddetails für Facebook-Anzeigen

Weitere Informationen zu den Entitäten und Felddetails findest du unter:

- Konto hinzufügen
- Kampagne
- Anzeigen-Set
- Anzeige
- Und kreativ
- Insight Ad-Konto
- Insights-Anzeigen
- Einblicke AdSets
- Insights-Kampagnen

Weitere Informationen finden Sie unter Marketing-API.

> **Note**
>
> Die Datentypen Struct und List werden in der Antwort der Konnektoren in den Datentyp String konvertiert.

Abfragen partitionieren

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiel für einen gültigen Wert:

```
"2022-01-01"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
FacebookADs_read = glueContext.create_dynamic_frame.from_options(
    connection_type="FacebookAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0",
        "PARTITION_FIELD": "created_time"
        "LOWER_BOUND": "2022-01-01"
        "UPPER_BOUND": "2024-01-02"
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Facebook-Anzeigen

Im Folgenden sind die Verbindungsoptionen für Facebook-Anzeigen aufgeführt:

- ENTITY_NAME(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Facebook-Anzeigen.
- API_VERSION(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Rest-API-Version von Facebook Ads, die Sie verwenden möchten. Zum Beispiel: v1.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `TRANSFER_MODE`(Zeichenfolge) — Standard: SYNC. Wird für asynchrones Lesen verwendet.

Einschränkungen und Hinweise für den Facebook Ads Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Facebook Ads Connector:

- Da Facebook Ads dynamische Metadaten unterstützt, können alle Felder abgefragt werden. Alle Felder unterstützen die Filterung und die Datensätze werden abgerufen, wenn die Daten verfügbar sind. Andernfalls gibt Facebook eine Antwort auf eine schlechte Anfrage (400) mit einer korrekten Fehlermeldung zurück.
- Die Anzahl der Anrufe einer App ist die Anzahl der Anrufe, die ein Benutzer innerhalb eines fortlaufenden einstündigen Zeitfensters tätigen kann, 200 multipliziert mit der Anzahl der Benutzer. Einzelheiten zu den Ratenlimits finden Sie unter Ratenlimits und Ratenlimits für geschäftliche Anwendungsfälle.

Verbindung zu Facebook Page Insights herstellen

Facebook-Seiten ermöglichen es Unternehmen und anderen Interessengruppen, Seiten für das soziale Netzwerk Facebook.com zu erstellen. Unternehmen nutzen diese Seiten, um Öffnungszeiten zu teilen, Ankündigungen zu machen und mit Kunden online in Kontakt zu treten. Wenn Sie ein Facebook Page Insights-Nutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Facebook Page Insights-Konto herstellen. Sie können Facebook Page Insights als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von Facebook Page Insights an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Facebook Page Insights
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Facebook Page Insights
- Konfiguration von Facebook Page Insights-Verbindungen
- Lesen aus Entitäten von Facebook Page Insights

- Verbindungsoptionen für Facebook Page Insights
- Einschränkungen und Hinweise für den Facebook Page Insights-Connector

AWS Glue Unterstützung für Facebook Page Insights

AWS Glue unterstützt Facebook Page Insights wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Facebook Page Insights abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte API-Versionen von Facebook Page Insights

Die folgenden API-Versionen von Facebook Page Insights werden unterstützt:

- v17
- v18
- v19
- v20
- v21

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
```

```
    {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Facebook Page Insights

Bevor Sie Daten von Facebook Page Insights übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Auf Facebook-Standardkonten wird direkt über Facebook zugegriffen.

- Zur Generierung des Zugriffstokens ist eine Benutzerauthentifizierung erforderlich.
- Der Facebook Page Insights-Konnektor implementiert den OAuth Benutzerzugriffstoken-Flow.
- Der Connector verwendet OAuth2 .0, um unsere API-Anfragen bei Facebook Page Insights zu authentifizieren. Dies fällt unter die Multi-Factor Authentication (MFA) -Architektur, die eine Obergruppe von 2FA darstellt. Es handelt sich um eine webbasierte Authentifizierung.
- Der Benutzer muss Berechtigungen für den Zugriff auf die Endgeräte erteilen. Für den Zugriff auf die Benutzerdaten erfolgt die Autorisierung der Endgeräte über Berechtigungen und Funktionen.

Konfiguration von Facebook Page Insights-Verbindungen

So konfigurieren Sie eine Facebook Page Insights-Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Facebook Page Insights aus.
   b. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   c. Wählen Sie die URL des Autorisierungscodes aus.
   d. Wählen Sie `secretName` die aus, in AWS Glue die Sie für diese Verbindung die Token einfügen möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen aus Entitäten von Facebook Page Insights

Voraussetzung

Ein Facebook-Seiten-Insights-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Inhalt der Seite	Ja	Nein	Ja	Ja	Ja
CTA-Klicks auf Seiten	Ja	Nein	Nein	Ja	Ja
Seitenint eraktion	Ja	Nein	Nein	Ja	Ja
Seitenimp ressionen	Ja	Nein	Nein	Ja	Ja
Beiträge von Seiten	Ja	Nein	Nein	Ja	Ja
Engagemen t nach Seitenbei trägen	Nein	Nein	Nein	Ja	Nein
Reaktionen auf Seitenbei träge	Nein	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Reaktionen auf Seiten	Ja	Nein	Nein	Ja	Ja
Geschichten	Ja	Nein	Nein	Ja	Ja
Demografische Daten der Seitenbenutzer	Ja	Nein	Nein	Ja	Ja
Videoansichten auf Seiten	Ja	Nein	Nein	Ja	Ja
Seitenansichten	Ja	Nein	Nein	Ja	Ja
Seite: Videobeiträge	Ja	Nein	Nein	Ja	Ja
Seiten	Nein	Ja	Nein	Ja	Nein
Feeds	Ja	Ja	Nein	Ja	Ja

Beispiel:

```
facebookPageInsights_read = glueContext.create_dynamic_frame. from options(
    connection_type="facebookpageinsights",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v21"
   }
```

Angaben zum Feld Facebook Page Insights:

Entität	Feld	Datentyp	Unterstützte Operatoren
Inhalt der Seite	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
CTA-Klicks auf die Seite	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Interaktion mit der Seite	Name	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Seitenimpressionen	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Beiträge auf der Seite	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Engagement nach Seitenbeiträgen	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Reaktionen auf Seitenbeiträge	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Demografische Daten der Seitenbenutzer	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Videoansichten von Seiten	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Seitenansichten	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Seite: Video-Beiträge	Name	String	N/A
	Intervall	Intervall	EQUAL_TO
	Seit	DateTime	EQUAL_TO
	Werte	Auflisten	N/A
	Title	String	N/A
	Beschreibung	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Seiten	Name	String	N/A
	Informationen	String	N/A
	Zugriffstoken	String	N/A
	ad_campaign	String	N/A
	Zugehörigkeit	String	N/A
	app_id	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Künstler, die wir mögen	String	N/A
	Kleidung	String	N/A
	Auszeichnungen	String	N/A
	band_interests	String	N/A
	band_mitglieder	String	N/A
	beste_Seite	String	N/A
	Biografie	String	N/A
	Geburtstag	String	N/A
	Buchungsagent	String	N/A
	Gebaut	String	N/A
	can_checkin	String	N/A
	kann_posten	String	N/A
	Kategorie	String	N/A
	kategorie_liste	Auflisten	N/A
	Check-ins	Ganzzahl	N/A
	Unternehmensübersicht	String	N/A
	verbundenes Instagram-Konto	String	N/A
	Kontaktadresse	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	country_page_likes	Ganzzahl	N/A
	Deckung	Struct	N/A
	kulinarisches Team	String	N/A
	derzeitiger Standort	String	N/A
	Informationen zur Liefer- und Abholopti on	Auflisten	N/A
	Beschreibung	String	N/A
	beschreibung_html	String	N/A
	anderes_offene_Ang ebote	Auflisten	N/A
	dirigiert_von	String	N/A
	Untertext anzeigen	String	N/A
	Antwortzeit der angezeigten Nachricht	String	N/A
	E-Mails	String	N/A
	Verlobung	String	N/A
	Anzahl der Fans	Ganzzahl	N/A
	Ausgewähltes Video	String	N/A
	Features	String	N/A
	Anzahl der Follower	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Essensstile	Auflisten	N/A
	gegründet	String	N/A
	general_info	String	N/A
	general_manager	String	N/A
	Genre	String	N/A
	Globaler_Name der Markenseite	String	N/A
	globale_marken_root_id	String	N/A
	hat_app_hinzugefügt_	Boolesch	N/A
	hat_auf_neue_Seitenerfahrung umgestellt	Boolesch	N/A
	hat_Whatsapp_Geschäftsnummer	Boolesch	N/A
	hat_whatsapp_nummer	Boolesch	N/A
	Heimatort	String	N/A
	Stunden	Struct	N/A
	Impressum	String	N/A
	Einflüsse	String	N/A
	instagram_business_account	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	ist_immer_geöffnet	Boolesch	N/A
	is_chain	Boolesch	N/A
	ist_Community-Seite	Boolesch	N/A
	ist_berechtigt_für _markeneigene Inhalte	Boolesch	N/A
	is_messenger_bot_get_started_enabled	Boolesch	N/A
	is_messenger_platform_bot	Boolesch	N/A
	ist im Besitz	Boolesch	N/A
	ist_dauer_geschlossen	Boolesch	N/A
	ist_veröffentlicht	Boolesch	N/A
	Name	String	N/A
	Aufgaben	Auflisten	N/A
	ist_nicht beansprucht	Boolesch	N/A
	is_webhooks_abonniert	Boolesch	N/A
	Leadgen_Tos_Acceptance_Time	DateTime	N/A
	leadgen_tos_accepted	Boolesch	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	leadgen_tos_accepting_user	String	N/A
	Leadgen_Tos_Accepting_User	Struct	N/A
	Link	Link	N/A
	Ort	Struct	N/A
	Mitglieder	String	N/A
	merchant_review_status	String	N/A
	messenger_ads_default_icebreakers	Auflisten	N/A
	messenger_ads_default_page_welcome_message	Struct	N/A
	messenger_ads_default_quick_replies	Auflisten	N/A
	Typ messenger_ads_quick_replies_	String	N/A
	Auftrag	String	N/A
	Mpg	String	N/A
	Name_mit_Standortdeskriptor	String	N/A
	Netzwerk	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	neue_like_count	Ganzzahl	N/A
	berechtigtes Angebot	Boolesch	N/A
	Gesamtbewertung mit Sternen	Gleitkommazahl	N/A
	Seitentoken	String	N/A
	übergeordnete Seite	String	N/A
	Parken	String	N/A
	Zahlungsoptionen	Struct	N/A
	persönliche_info	String	N/A
	persönliche_Interessen	String	N/A
	Pharma_Sicherheitsinformationen	String	N/A
	Phone	String	N/A
	Abholoptionen	Auflisten	N/A
	ort_typ	String	N/A
	plot_outline	String	N/A
	presse_kontakt	String	N/A
	Preisspanne	String	N/A
	URL mit Datenschutzinformationen	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	produziert_von	String	N/A
	Produkte	String	N/A
	förderfähig	Boolesch	N/A
	Grund für die Beförderung nicht förderfähig	String	N/A
	öffentlicher Nahverkehr	String	N/A
	Anzahl der Bewertungen	Ganzzahl	N/A
	Datensatzbezeichnung	String	N/A
	Veröffentlichungsdatum	String	N/A
	Restaurant-Dienstleistungen	Struct	N/A
	Restaurantspezialitäten	Struct	N/A
	Plan	String	N/A
	screenplay_von	String	N/A
	Saison	String	N/A
	einzeilige Adresse	String	N/A
	In den Hauptrollen	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	start_info	Struct	N/A
	Speichercode	String	N/A
	store_location_descriptor	String	N/A
	Filialnummer	Ganzzahl	N/A
	Studio	String	N/A
	unterstützt Donate_Button_in_Livevideo	Boolesch	N/A
	wir reden über_count	Ganzzahl	N/A
	temporäre_Status	String	N/A
	Anzahl ungelesener Nachrichten	Ganzzahl	N/A
	Anzahl ungelesener Nachrichten	Ganzzahl	N/A
	Anzahl ungesehener Nachrichten	Ganzzahl	N/A
	Username	String	N/A
	Status der Überprüfung	String	N/A
	voip_info	Struct	N/A
	Website	String	N/A
	were_hier_zählen	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	whatsapp_nummer	String	N/A
geschrieben von	String	N/A	
Feeds	Id	String	N/A
	Aktionen	Auflisten	N/A
	admin_creator	Object	N/A
	Anwendung	Object	N/A
	Anlagen	Objekte	N/A
	zurückdatierte_Zeit	DateTime	N/A
	Aufruf zum Handeln	Object	N/A
	kann_privat antworten	Boolesch	N/A
	Dateianhänge für Kinder	Auflisten	N/A
	Koordinaten	Struct	N/A
	Erstellungszeit	DateTime	N/A
	Ereignis	Struct	N/A
	erweiterte_Höhe	Ganzzahl	N/A
	erweiterte_Breite	Ganzzahl	N/A
	Feed-Targeting	Object	N/A
	Aus	Object	N/A
	volles_Bild	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Höhe	Ganzzahl	N/A
	Symbol	String	N/A
	Eignung für Instagram	String	N/A
	ist_beförderungswürdig_	Boolesch	N/A
	ist_abgelaufen	Boolesch	N/A
	ist_versteckt	Boolesch	N/A
	ist_inline_erstellt	Boolesch	N/A
	ist_instagram_berechtigt	Boolesch	N/A
	ist_beliebt	Boolesch	N/A
	ist_veröffentlicht	Boolesch	N/A
	ist_sphärisch	Boolesch	N/A
	Fehlermeldung	String	N/A
	Nachrichten-Tags	Auflisten	N/A
	multi_share_end_card	Boolesch	N/A
	multi_share_optimiert	Boolesch	N/A
	Eltern-ID	String	N/A
	Permalink-URL	String	N/A
	Place	String	N/A
	Privacy (Datenschutz)	Object	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	beworbene_ID	String	N/A
	Status der Werbeaktion	String	N/A
	Eigenschaften	Auflisten	N/A
	Geplanter Veröffentlichungszeitpunkt	Gleitkommazahl	N/A
	Aktien	Object	N/A
	status_type	String	N/A
	Geschichte	String	N/A
	story_tags	Auflisten	N/A
	Abonniert	Boolesch	N/A
	Ziel	Struct	N/A
	Targeting	Object	N/A
	Bis	Object	N/A
	timeline_visibility	String	N/A
	aktualisierte_Uhrzeit	DateTime	N/A
	Via	Struct	N/A
	Eignung für den Kauf von Videos	Auflisten	N/A
	Width	Ganzzahl	N/A
	Seit	DateTime	EQUAL_TO

Abfragen partitionieren

Filterbasierte Partitionierung:

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
facebookPageInsights_read = glueContext.create_dynamic_frame.from_options(
     connection_type="facebookpageinsights",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v21",
         "PARTITION_FIELD": "created_Time"
         "LOWER_BOUND": "2024-10-27T07:00:00+0000"
         "UPPER_BOUND": "2024-10-27T07:00:00+0000"
         "NUM_PARTITIONS": "10"
     }
```

Verbindungsoptionen für Facebook Page Insights

Im Folgenden sind Verbindungsoptionen für Facebook Page Insights aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Facebook Page Insights.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Rest-API-Version von Facebook Page Insights, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige URL der Facebook-Seite Insights-Instanz.

Einschränkungen und Hinweise für den Facebook Page Insights-Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Facebook Page Insights-Connector:

- Die meisten Metriken werden alle 24 Stunden aktualisiert.
- Es sind nur Insights-Daten der letzten zwei Jahre verfügbar.
- Bei Verwendung der `until` Parameter `since` und können nur Erkenntnisse aus 90 Tagen gleichzeitig angezeigt werden.

Verbindung zu Freshdesk herstellen

Freshdesk ist eine cloudbasierte Kundensupport-Software, die sowohl funktionsreich als auch benutzerfreundlich ist. Da mehrere Support-Kanäle verfügbar sind, darunter Live-Chat, E-Mail, Telefon und soziale Medien, können Sie Kunden über ihre bevorzugte Kommunikationsmethode helfen. Wenn Sie ein Freshdesk-Benutzer sind, können Sie sich mit Ihrem Freshdesk-Konto verbinden AWS Glue . Sie können Freshdesk als Datenquelle in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von Freshdesk an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Freshdesk
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Freshdesk konfigurieren
- Konfiguration von Freshdesk-Verbindungen
- Aus Freshdesk-Entitäten lesen
- Freshdesk-Verbindungsoptionen
- Einschränkungen und Hinweise für Freshdesk Connector

AWS Glue Unterstützung für Freshdesk

AWS Glue unterstützt Freshdesk wie folgt:

Wird es als Quelle unterstützt?

Ja — Sync und Async. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Freshdesk abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Freshdesk-API-Versionen

Die folgenden Freshdesk-API-Versionen werden unterstützt:

- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
     ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben

Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Freshdesk konfigurieren

Bevor Sie Daten von Freshdesk AWS Glue übertragen können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Ein Freshdesk-Konto. Sie können zwischen den Editionen Free, Growth, Pro oder Enterprise wählen.
- Der API-Schlüssel eines Freshdesk-Benutzers.

Konfiguration von Freshdesk-Verbindungen

Freshdesk unterstützt benutzerdefinierte Authentifizierung.

Die öffentliche Freshdesk-Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie unter Freshdesk-Authentifizierung.

Im Folgenden finden Sie die Schritte zur Konfiguration der Freshdesk-Verbindung:

- Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
  - Bei einer vom Kunden verwalteten verbundenen App sollte das Geheimnis den API-Schlüssel der verbundenen App mit einem `apiKey` Schlüssel enthalten. Beachten Sie, dass Sie pro Verbindung in ein Geheimnis erstellen müssen AWS Glue.
- Stellen Sie im unter Datenverbindungen eine Verbindung her AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
  - Wählen Sie bei der Auswahl einer Datenquelle Freshdesk aus.
  - Geben Sie die `INSTANCE_URL` Freshdesk-Instanz an, zu der Sie eine Verbindung herstellen möchten.
  - Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
```

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

  - Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.
  - Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
- Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
- Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Freshdesk-Entitäten lesen

Voraussetzung

Ein Freshdesk-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für Sync-Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kundendienstmitarbeiters (Kundendi	Ja	Ja	Nein	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
enstmitarbeiter)					
Geschäftszeiten	Nein	Ja	Nein	Ja	Ja
Unternehmen	Ja	Ja	Nein	Ja	Ja
Kontakte	Ja	Ja	Nein	Ja	Ja
Konversationen	Nein	Ja	Nein	Ja	Nein
E-Mail-Konfigurationen	Nein	Ja	Nein	Ja	Nein
E-Mail-Posteingänge	Ja	Ja	Ja	Ja	Nein
Kategorien des Forums	Nein	Ja	Nein	Ja	Nein
Foren	Nein	Ja	Nein	Ja	Nein
Gruppen	Nein	Ja	Nein	Ja	Nein
Produkte	Nein	Ja	Nein	Ja	Nein
Rollen	Nein	Ja	Nein	Ja	Nein
Zufriedenheitsbewertungen	Ja	Ja	Nein	Ja	Nein
Fähigkeiten	Nein	Ja	Nein	Ja	Nein
Lösungen	Ja	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Umfragen	Nein	Ja	Nein	Ja	Nein
Eintrittskarten	Ja	Ja	Ja	Ja	Ja
Zeiteinträge	Ja	Ja	Nein	Ja	Nein
Themen	Nein	Ja	Nein	Ja	Nein
Kommentare zum Thema	Nein	Ja	Nein	Ja	Nein

Unterstützte Entitäten für die asynchrone Quelle:

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Unternehmen	v2	Nein	Nein	Nein	Nein	Nein
Kontakte	v2	Nein	Nein	Nein	Nein	Nein

Beispiel:

```
freshdesk_read = glueContext.create_dynamic_frame.from_options(
    connection_type="freshdesk",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v2"
    }
```

Angaben zur Einheit und zum Feld von Freshdesk:

Entität	Feld
Kundendienstmitarbeiters (Kundendienstmitarbeiter)	https://developers.freshdesk.com/api/#list_all_agents
Geschäftszeiten	https://developers.freshdesk.com/api/#list_all_business_hours
Kommentare	https://developers.freshdesk.com/api/#comment_attributess
Unternehmen	https://developers.freshdesk.com/api/#companies
Kontakte	https://developers.freshdesk.com/api/#list_all_contacts
Konversationen	https://developers.freshdesk.com/api/#list_all_ticket_notes
E-Mail-Konfigurationen	https://developers.freshdesk.com/api/#list_all_email_configs
E-Mail-Posteingänge	https://developers.freshdesk.com/api/#list_all_email_mailboxes
Forum-Kategorien	https://developers.freshdesk.com/api/#category_attributes
Foren	https://developers.freshdesk.com/api/#forum_attributes
Gruppen	https://developers.freshdesk.com/api/#list_all_groups
Produkte	https://developers.freshdesk.com/api/#list_all_products

Entität	Feld
Rollen	https://developers.freshdesk.com/api/#list_all_roles
Zufriedenheitsbewertung	https://developers.freshdesk.com/api/#view_all_satisfaction_ratingss
Fähigkeiten	https://developers.freshdesk.com/api/#list_all_skills
Lösungen	https://developers.freshdesk.com/api/#solution_content
Umfragen	https://developers.freshdesk.com/api/#list_all_survey
Eintrittskarten	https://developers.freshdesk.com/api/#list_all_tickets
Zeiteinträge	https://developers.freshdesk.com/api/#list_all_time_entries
Themen	https://developers.freshdesk.com/api/#topic_attributes

Abfragen partitionieren

Filterbasierte Partitionierung:

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

Beispiele für gültige Werte:

```
"2024-09-30T01:01:01.000Z"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
freshDesk_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshdesk",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v2",
         "PARTITION_FIELD": "Created_Time"
         "LOWER_BOUND": " 2024-10-27T23:16:08Z“
         "UPPER_BOUND": " 2024-10-27T23:16:08Z"
         "NUM_PARTITIONS": "10"
     }
```

Freshdesk-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Freshdesk aufgeführt:

- ENTITY_NAME(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Freshdesk.
- API_VERSION(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Freshdesk Rest API-Version, die Sie verwenden möchten.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige Freshdesk-Instanz-URL.
- `TRANSFER_MODE`(Zeichenfolge) — Wird verwendet, um anzugeben, ob die Art der Verarbeitung `SYNC` standardmäßig auf „Gefällt“ `SYNC` oder eingestellt `ASYNC` ist. (Optional)

Einschränkungen und Hinweise für Freshdesk Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Freshdesk-Connector:

- Für die `Tickets` Entitäten `CompanyContacts`, und mit Filterung gelten Einschränkungen bei der Seitennummerierung. Sie geben nur 30 Datensätze pro Seite zurück, und der Seitenwert kann auf maximal 10 festgelegt werden (es werden maximal 300 Datensätze abgerufen).
- Die `Tickets` Entität ruft keine Datensätze ab, die älter als 30 Tage sind.
- Die `Tickets` Entitäten `CompanyContacts`, und unterstützen den Datentyp „Datum“ bei der Filterung. Sie sollten für diese drei Entitäten die Triggerfrequenzen „Täglich“ auswählen. Die Auswahl von „Minuten“ oder „Stündlich“ kann zu doppelten Daten führen. Außerdem sollte bei der Auswahl dieser Felder für die Filterung nur der Datumswert ausgewählt werden, da nur der Datumsanteil des ausgewählten Zeitstempels berücksichtigt wird.
- Die Anzahl der API-Aufrufe pro Minute richtet sich nach Ihrem Plan. Dieses Limit gilt für das gesamte Konto, unabhängig von Faktoren wie der Anzahl der Agenten oder den IP-Adressen, die für die Anrufe verwendet wurden. Für alle Testbenutzer gibt es ein Standard-API-Limit von 50 Anrufen/Minute. Weitere Informationen finden Sie auf Freshdesk
- Für jede Entität wird jeweils nur ein Export/Async-Job verarbeitet. Ein neuer Job wird erst verarbeitet, wenn der bestehende Job erfolgreich abgeschlossen wurde oder fehlgeschlagen ist. Weitere Informationen finden Sie auf Freshdesk
- Die folgenden Felder werden für Sync-API-Aufrufe unterstützt, dürfen aber nicht im Async-API-Anfragetext übergeben werden.

- id
- created_at
- updated_at
- updated_since
- aktiv
- firmen_id
- andere_Unternehmen
- Avatar
- alle_Tickets ansehen
- deleted
- andere_E-Mails
- state
- Tag (Markierung)
- tags

Verbindung zu Freshsales herstellen

Freshsales ist ein intuitives CRM, das Vertriebsmitarbeitern hilft, dem Rätselraten im Vertrieb ein Ende zu setzen. Dank der integrierten Funktionen für Telefon und E-Mail, Aufgaben, Termine und Notizen müssen Vertriebsmitarbeiter nicht zwischen den Tabs hin- und herschalten, um potenzielle Kunden zu verfolgen. Mit der Pipeline-Ansicht können Sie Ihre Geschäfte besser verwalten und mehr Geschäfte zum Abschluss bringen. Wenn Sie ein Freshsales-Nutzer sind, können Sie sich mit Ihrem AWS Glue Freshsales-Konto verbinden. Sie können Freshsales als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von Freshsales an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Freshsales
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Freshsales konfigurieren
- Konfiguration von Freshsales-Verbindungen
- Lesen von Freshsales-Einheiten
- Freshsales-Verbindungsoptionen

- Einschränkungen bei Freshsales

AWS Glue Unterstützung für Freshsales

AWS Glue unterstützt Freshsales wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Freshsales abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Freshsales-API-Versionen

Die folgenden Freshsales API-Versionen werden unterstützt:

- v1.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
```

```
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Freshsales konfigurieren

Bevor Sie Daten von Freshsales AWS Glue übertragen können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Freshsales-Konto.
- Sie haben einen Benutzer-API-Schlüssel.

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem AWS Glue Freshsales-Konto verbinden. Für typische Verbindungen müssen Sie in Freshsales nichts weiter tun.

Konfiguration von Freshsales-Verbindungen

Freshsales unterstützt benutzerdefinierte Authentifizierung.

Die öffentliche Freshsales-Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Authentifizierung finden Sie unter Authentifizierung.

So konfigurieren Sie eine Freshsales-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret den API-Schlüssel der verbundenen App mit einem *apiSecretKey* Schlüssel enthalten. Das Secret muss außerdem ein weiteres Schlüssel-Wert-Paar mit *apiKey* als Schlüssel und *token* als Wert enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen. AWS Glue

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Freshsales aus.
   b. Geben Sie das `INSTANCE_URL` Freshsales-Konto an, mit dem Sie sich verbinden möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen von Freshsales-Einheiten

Voraussetzung

Ein Freshsales-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Konten	Ja	Ja	Ja	Ja	Ja
Kontakte	Ja	Ja	Ja	Ja	Ja

Beispiel:

```
freshSales_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshsales",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v1.0"
     }
```

Angaben zur Einheit und zum Feld von Freshsales:

Freshsales bietet Endpunkte zum dynamischen Abrufen von Metadaten für unterstützte Entitäten. Dementsprechend wird die Bedienerunterstützung auf der Datentypebene erfasst.

Entität	Datentyp	Unterstützte Operatoren
Freshsale-Einheiten (alle)	Ganzzahl	! =, =, <, <=, >, >=, ZWISCHEN

Entität	Datentyp	Unterstützte Operatoren
	String	Wie, =,! =
	BigInteger	! =, =, <, <=, >, >=, ZWISCHEN
	Boolesch	=
	Double	! =, =, <, <=, >, >=, ZWISCHEN
	BigDecimal	! =, =, <, <=, >, >=, ZWISCHEN
	Datum	! =, =, <, <=, >, >=, ZWISCHEN
	DateTime	! =, =, <, <=, >, >=, ZWISCHEN
	Struct	N/A
	Auflisten	N/A

Abfragen partitionieren

Filterbasierte Partitionierung:

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,, und angeben `LOWER_BOUND``UPPER_BOUND`, `NUM_PARTITIONS` ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir den Wert im ISO-Format.

Beispiele für gültige Werte:

```
"2024-09-30T01:01:01.000Z"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
freshSales_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshsales",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v1",
         "PARTITION_FIELD": "Created_Time"
         "LOWER_BOUND": " 2024-10-15T21:16:25Z"
         "UPPER_BOUND": " 2024-10-20T21:25:50Z"
         "NUM_PARTITIONS": "10"
     }
```

Freshsales-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Freshsales aufgeführt:

- ENTITY_NAME(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Freshsales.
- API_VERSION(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Freshsales Rest API-Version, die Sie verwenden möchten.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.

- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `INSTANCE_URL`(Zeichenfolge) — Wird zum Lesen verwendet. Eine gültige Freshsales-Instanz-URL.

Einschränkungen bei Freshsales

Im Folgenden finden Sie Einschränkungen oder Hinweise für Freshsales:

- In Freshsales liegt das API-Ratenlimit bei 1000 API-Anfragen pro Stunde und Konto (siehe Fehler). Dieses Limit kann jedoch mit dem Enterprise-Abonnement erweitert werden (siehe Tarifvergleich).

Verbindung zu Google Ads herstellen

Die Google Ads API ist die programmatische Schnittstelle zu Google Ads, die für die Verwaltung großer oder komplexer Google Ads-Konten und -Kampagnen verwendet wird. Wenn Sie ein Google Ads-Nutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Google Ads-Konto herstellen. Anschließend können Sie Google Ads als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Google Ads und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Google Ads
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Google Ads konfigurieren
- Google Ads-Verbindungen konfigurieren
- Aus Google Ads-Entitäten lesen
- Verbindungsoptionen für Google Ads
- Ein Google Ads-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Google Ads

AWS Glue unterstützt Google Ads wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Google Ads abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Google Ads API-Versionen

v18

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Google Ads konfigurieren

Bevor Sie die Übertragung von Google Ads verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Google Ads-Konto mit E-Mail und Passwort. Weitere Informationen zum Erstellen eines Kontos finden Sie unter Google Ads-Konto erstellen.
- Ihr Google Ads-Konto ist für den API-Zugriff aktiviert. Jegliche Nutzung der Google Ads API ist ohne zusätzliche Kosten möglich.
- Mit Ihrem Google Ads-Konto können Sie verbundene Apps installieren. Wenn Sie keinen Zugriff auf diese Funktion haben, wenden Sie sich an Ihren Google Ads-Administrator.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Google Ads-Konto herstellen.

Google Ads-Verbindungen konfigurieren

Google Ads unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für OAuth2.

Diese Art der Gewährung gilt als „dreibeiniges“, OAuth da sie darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Nutzer zu Google Ads weiter, wo er sich anmelden und den

angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Google Ads-Instanz gewähren muss.

Nutzer können sich dafür entscheiden, ihre eigene verbundene App in Google Ads zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben. In diesem Szenario werden sie weiterhin zu Google Ads weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken.
Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.

Weitere Informationen finden Sie in der öffentlichen Google Ads-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss.

So konfigurieren Sie eine Google Ads-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für den AuthorizationCode Zuschusstyp:
      - Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Google Ads aus.
   b. Geben Sie `developer token` die Google Ads an, mit denen Sie eine Verbindung herstellen möchten.
   c. Geben Sie `MANAGER ID` die Google Ads an, wenn Sie sich als Manager anmelden möchten.
   d. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

e. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

f. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Google Ads-Entitäten lesen

Voraussetzungen

- Ein Google Ads-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Anzeigengruppe Anzeige	Ja	Ja	Ja	Nein	Ja
Anzeigengruppe	Ja	Ja	Ja	Nein	Ja
Kampagnenbudget	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Kontobudget	Ja	Nein	Ja	Ja	Nein
Kampagne	Ja	Ja	Ja	Ja	Ja
Account	Ja	Nein	Ja	Nein	Nein

Beispiel

```
googleAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googleads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "campaign-3467***",
        "API_VERSION": "v16"
    }
```

Entitäts- und Felddetails in Google Ads

Entität	Feld	Datentyp	Unterstützte Operatoren
Account	RessourcenNamen	String	!=, =
Account	callReportingEnabled	Boolesch	!=, =
Account	callConversionReportingAktiviert	Boolesch	!=, =
Account	callConversionAction	String	!=, =
Account	conversionTrackingId	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	crossAccountConversionTrackingId	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Account	payPerConversionEligibilityFailureReasons	Auflisten	
Account	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	currencyCode	String	! =, =, WIE
Account	Zeitzone	String	! =, =, WIE
Account	autoTaggingEnabled	Boolesch	!=, =
Account	hasPartnersBadge	Boolesch	!=, =
Account	manager	Boolesch	!=, =
Account	Testkonto	Boolesch	!=, =
Account	date	Datum	ZWISCHEN, =, <, >, <=, >=
Account	CostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	acceptedCustomerDataBedingungen	Boolesch	
Account	conversionTrackingStatus	String	! =, =, WIE
Account	enhancedConversionsForLeadsEnabled	Boolesch	
Account	googleAdsConversionKunde	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
Account	Status	String	!=, =
Account	allConversionsByConversionDate	Double	!=, =, <, >
Account	allConversionsValueByConversionDate	Double	!=, =, <, >
Account	conversionsByConversionDatum	Double	!=, =, <, >
Account	conversionsValueByConversionDate	Double	!=, =, <, >
Account	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Account	Videoansichten	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	Ungültige Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	costPerAllKonvertierungen	Double	!=, =, <, >
Account	costPerConversion	Double	!=, =, <, >
Account	Konvertierungen	Double	!=, =, <, >
Account	absoluteTopImpressionProzentsatz	Double	!=, =, <, >

Entität	Feld	Datentyp	Unterstützte Operatoren
Account	Impressionen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	topImpressionPercentage	Double	!=, =, <, >
Account	Durchschnittlicher CPC	Double	!=, =, <, >
Account	activeViewMeasurableCostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Account	Durchschnittliche Kosten	Double	!=, =, <, >
Account	ctr	Double	!=, =, <, >
Account	activeViewCtr	Double	!=, =, <, >
Account	searchImpressionShare	Double	!=, =, <, >
Account	Konvertierungsaktion	String	!=, =
Account	conversionActionCategory	String	!=, =
Account	conversionActionName	String	! =, =, WIE
Konto, Budget	RessourcenNamen	String	!=, =
Kontobudget	Status	String	!=, =
Kontobudget	proposedEndTimeTyp	String	!=, =
Kontobudget	approvedEndTimeTyp	String	!=, =

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontobudget	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kontobudget	Einrichtung der Abrechnung	String	!=, =
Kontobudget	Name	String	! =, =, WIE
Konto, Budget	approvedStartDateZeit	DateTime	ZWISCHEN, =, <, >, <=, >=
Konto, Budget	proposedSpendingLimitMikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kontobudget	approvedSpendingLimitMikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kontobudget	adjustedSpendingLimitMikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kontobudget	amountServedMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	RessourcenNamen	String	! =, =, WIE
Anzeigengruppe	Status	String	! =, =, WIE
Anzeigengruppe	Typ	String	! =, =, WIE
Anzeigengruppe	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	Name	String	! =, =, WIE
Anzeigengruppe	Kampagne	String	!=, =

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeigengruppe	cpcBidMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	targetCpaMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	cpmBidMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	cpvBidMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	targetCpmMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	effectiveTargetCpa Mikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	date	Datum	ZWISCHEN, =, <, >, <=, >=
Anzeigengruppe	CostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	useAudienceGrouped	Boolesch	!=, =
Anzeigengruppe	effectiveCpcBidMikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	allConversionsByCo nversionDate	Double	!=, =, <, >
Anzeigengruppe	allConversionsValu eByConversionDate	Double	!=, =, <, >
Anzeigengruppe	conversionsByConve rsionDatum	Double	!=, =, <, >

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeigengruppe	conversionsValueByConversionDate	Double	!=, =, <, >
Anzeigengruppe	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Anzeigengruppe	valuePerConversionsByConversionDate	Double	!=, =, <, >
Anzeigengruppe	Durchschnittliche Kosten	Double	!=, =, <, >
Anzeigengruppe	costPerAllKonvertierungen	Double	!=, =, <, >
Anzeigengruppe	costPerConversion	Double	!=, =, <, >
Anzeigengruppe	averagePageViews	Double	!=, =, <, >
Anzeigengruppe	Videoansichten	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe	Alle Konvertierungen	Double	!=, =, <, >
Anzeigengruppe	Durchschnittlicher CPC	Double	!=, =, <, >
Anzeigengruppe	absoluteTopImpressionProzentsatz	Double	!=, =, <, >
Anzeigengruppe	Impressionen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeigengruppe	topImpressionPercentage	Double	!=, =, <, >
Anzeigengruppe	activeViewCtr	Double	!=, =, <, >
Anzeigengruppe	ctr	Double	!=, =, <, >
Gruppe hinzufügen	searchTopImpressionTeilen	Double	!=, =, <, >
Anzeigengruppe	searchImpressionShare	Double	!=, =, <, >
Anzeigengruppe	searchAbsoluteTopImpressionShare	Double	!=, =, <, >
Anzeigengruppe	Relative CTR	Double	!=, =, <, >
Anzeigengruppe	Konvertierungsaktion	String	!=, =
Anzeigengruppe	conversionActionCategory	String	!=, =
Anzeigengruppe	conversionActionName	String	! =, =, WIE
Anzeigengruppe	Maske aktualisieren	String	
Anzeigengruppe	create	Struct	
Anzeigengruppe	update	Struct	
Anzeigengruppe	Primärer Status	String	!=, =
Anzeigengruppe	primaryStatusReasons	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeigengruppe Anzeige	RessourcenNamen	String	!=, =
Anzeige der Anzeigengruppe	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	Status	String	!=, =
Anzeige der Anzeigengruppe	labels	Auflisten	
Anzeige der Anzeigengruppe	Anzeigengruppe	String	!=, =
Anzeigengruppe Anzeige	CostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	Genehmigungsstatus	String	!=, =
Anzeige der Anzeigengruppe	Status überprüfen	String	!=, =
Anzeige der Anzeigengruppe	Stärke der Werbung	String	!=, =
Anzeige der Anzeigengruppe	Typ	String	!=, =
Anzeige der Anzeigengruppe	Name des Unternehmens	String	! =, =, WIE
Anzeige, Gruppe, Anzeige	date	Datum	ZWISCHEN, =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeigengruppe > Anzeige	allConversionsByConversionDate	Double	!=, =, <, >
Anzeige der Anzeigengruppe	allConversionsValueByConversionDate	Double	!=, =, <, >
Anzeige der Anzeigengruppe	conversionsByConversionDatum	Double	!=, =, <, >
Anzeige Gruppe Anzeige	conversionsValueByConversionDate	Double	!=, =, <, >
Anzeige der Anzeigengruppe	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Anzeige der Anzeigengruppe	valuePerConversionsByConversionDate	Double	!=, =, <, >
Anzeige der Anzeigengruppe	activeViewMeasurableCostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	Durchschnittliche Kosten	Double	!=, =, <, >
Anzeige der Anzeigengruppe	costPerAllKonvertierungen	Double	!=, =, <, >
Anzeigengruppe Anzeige	costPerConversion	Double	!=, =, <, >
Anzeige der Anzeigengruppe	Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	averagePageViews	Double	!=, =, <, >

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeige der Anzeigengruppe	Videoansichten	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	Alle Konvertierungen	Double	!=, =, <, >
Anzeigengruppe Anzeige	Durchschnittlicher CPC	Double	!=, =, <, >
Anzeige der Anzeigengruppe	topImpressionPercentage	Double	!=, =, <, >
Anzeige der Anzeigengruppe	Impressionen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Anzeigengruppe: Anzeige	absoluteTopImpressionProzentsatz	Double	!=, =, <, >
Anzeigengruppe Anzeige	activeViewCtr	Double	!=, =, <, >
Anzeige der Anzeigengruppe	ctr	Double	!=, =, <, >
Anzeigengruppe Anzeige	Konvertierungsaktion	String	!=, =
Anzeigengruppe Anzeige	conversionActionCategory	String	!=, =
Anzeige der Anzeigengruppe	conversionActionName	String	! =, =, WIE
Anzeige, Gruppe, Anzeige	Maske aktualisieren	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
Anzeige der Anzeigengruppe	create	Struct	
Anzeige der Anzeigengruppe	update	Struct	
Anzeige der Anzeigengruppe	policyValidationParameter	Struct	
Anzeige der Anzeigengruppe	Primärer Status	String	!=, =
Anzeigengruppe Anzeige	primaryStatusReasons	Auflisten	
Kampagne	RessourcenNamen	String	!=, =
Kampagne	Status	String	!=, =
Kampagne	Basiskampagne	String	!=, =
Kampagne	Name	String	! =, =, WIE
Kampagne	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	Kampagnenbudget	String	! =, =, WIE
Kampagne	startDate	Datum	ZWISCHEN, =, <, >, <=, >=
Kampagne	endDate	Datum	ZWISCHEN, =, <, >, <=, >=
Kampagne	adServingOptimizationStatus	String	!=, =

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagne	advertisingChannel Type	String	!=, =
Kampagne	advertisingChannel SubTyp	String	!=, =
Kampagne	Art des Experiments	String	!=, =
Kampagne	Status des Servierens	String	!=, =
Kampagne	biddingStrategyType	String	!=, =
Kampagne	domainName	String	! =, =, WIE
Kampagne	languageCode	String	! =, =, WIE
Kampagne	useSuppliedUrlsNur	Boolesch	!=, =
Kampagne	positiveGeoTargetTyp	String	!=, =
Kampagne	negativeGeoTargetT yp	String	!=, =
Kampagne	Zahlungsmodus	String	!=, =
Kampagne	optimizationGoalTy pes	Auflisten	
Kampagne	date	Datum	ZWISCHEN, =, <, >, <=, >=
Kampagne	Durchschnittliche Kosten	Double	
Kampagne	Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagne	CostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	Impressionen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	useAudienceGrouped	Boolesch	!=, =
Kampagne	activeViewMeasurableCostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	costPerAllKonvertierungen	Double	!=, =, <, >
Kampagne	costPerConversion	Double	!=, =, <, >
Kampagne	Ungültige Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	publisherPurchasedClicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	averagePageViews	Double	!=, =, <, >
Kampagne	Videoansichten	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagne	allConversionsByConversionDate	Double	!=, =, <, >
Kampagne	allConversionsValueByConversionDate	Double	!=, =, <, >
Kampagne	conversionsByConversionDatum	Double	!=, =, <, >

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagne	conversionsValueByConversionDate	Double	!=, =, <, >
Kampagne	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Kampagne	valuePerConversionsByConversionDate	Double	!=, =, <, >
Kampagne	Alle Konvertierungen	Double	!=, =, <, >
Kampagne	absoluteTopImpressionProzentsatz	Double	!=, =, <, >
Kampagne	searchAbsoluteTopImpressionShare	Double	!=, =, <, >
Kampagne	Durchschnittlicher CPC	Double	!=, =, <, >
Kampagne	searchImpressionShare	Double	!=, =, <, >
Kampagne	searchTopImpressionTeilen	Double	!=, =, <, >
Kampagne	activeViewCtr	Double	!=, =, <, >
Kampagne	ctr	Double	!=, =, <, >
Kampagne	Relative CTR	Double	!=, =, <, >
Kampagne	Maske aktualisieren	String	
Kampagne	create	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagne	update	Struct	
Kampagnenbudget	RessourcenNamen	String	!=, =
Kampagnenbudget	id	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	Status	String	!=, =
Kampagnenbudget	Methode der Lieferung	String	!=, =
Kampagnenbudget	Zeitraum	String	!=, =
Kampagnenbudget	Typ	String	!=, =
Kampagnenbudget	Name	String	! =, =, WIE
Kampagnenbudget	Anzahl Mikros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	Explizit geteilt	Boolesch	!=, =
Kampagnenbudget	Anzahl der Referenzen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	hasRecommendedBudget	Boolesch	!=, =
Kampagnenbudget	date	Datum	ZWISCHEN, =, <, >, <=, >=
Kampagnenbudget	CostMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	startDate	Datum	ZWISCHEN, =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagnenbudget	endDate	Datum	ZWISCHEN, =, <, >, <=, >=
Kampagnenbudget	maximizeConversionValueTargetRoas	Double	!=, =, <, >
Kampagnenbudget	maximizeConversionsTargetCpaMicros	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	selectiveOptimizationConversionAktionen	String	
Kampagnenbudget	Durchschnittliche Kosten	Double	!=, =, <, >
Kampagnenbudget	costPerAllKonversionen	Double	!=, =, <, >
Kampagnenbudget	costPerConversion	Double	!=, =, <, >
Kampagnenbudget	Videoansichten	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	Klicks	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=
Kampagnenbudget	Alle Konversionen	Double	!=, =, <, >
Kampagnenbudget	valuePerAllKonversionen	Double	!=, =, <, >
Kampagnenbudget	Durchschnittlicher CPC	Double	!=, =, <, >
Kampagnenbudget	Impressionen	BigInteger	ZWISCHEN, =,! =, <, >, <=, >=

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagnenbudget	ctr	Double	!=, =, <, >
Kampagnenbudget	Maske aktualisieren	String	
Kampagnenbudget	create	Struct	
Kampagnenbudget	update	Struct	

Abfragen partitionieren

Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:`"2024-02-06"`.
- `UPPER_BOUND`: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt.

Entity Name	Feld partitionieren	Datentyp
Anzeigengruppe Anzeige	date	Datum
Anzeigengruppe	date	Datum
Kampagne	date	Datum

Entity Name	Feld partitionieren	Datentyp
Kampagnenbudget	date	Datum

Beispiel

```
googleads_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googleads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "campaign-3467***",
        "API_VERSION": "v16",
        "PARTITION_FIELD": "date"
        "LOWER_BOUND": "2024-01-01"
        "UPPER_BOUND": "2024-06-05"
        "NUM_PARTITIONS": "10"
    }
)
```

Verbindungsoptionen für Google Ads

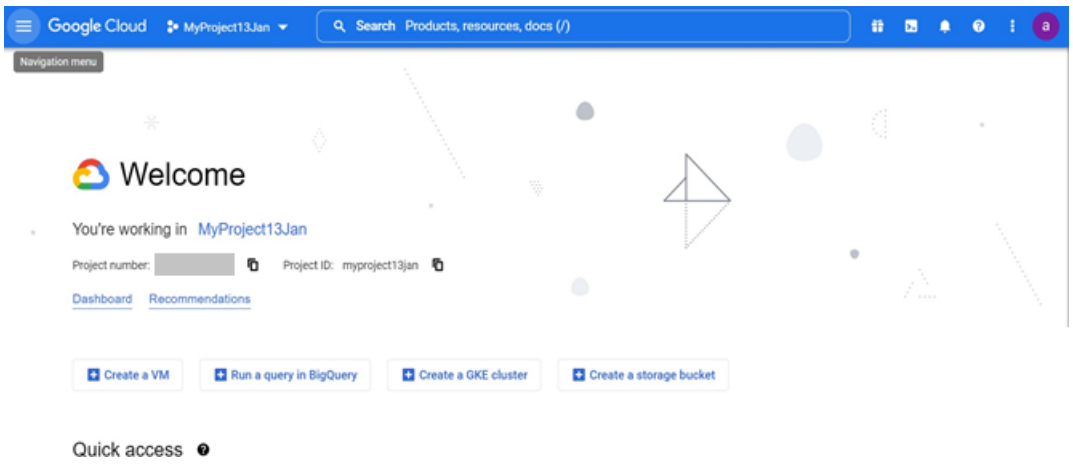
Im Folgenden sind die Verbindungsoptionen für Google Ads aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Google Ads.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Die Rest-API-Version von Google Ads, die Sie verwenden möchten. Beispiel: v16.
- `DEVELOPER_TOKEN`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Erforderlich, um den Entwickler oder die Anwendung zu authentifizieren, die Anfragen an die API stellen.
- `MANAGER_ID`(Zeichenfolge) — Wird für Lesen/Schreiben verwendet. Eine eindeutige Kennung, mit der Sie mehrere Google Ads-Konten verwalten können. Dies ist die Kunden-ID des autorisierten Managers. Wenn Ihr Zugriff auf das Kundenkonto über ein Verwaltungskonto erfolgt, `MANAGER_ID` ist das erforderlich. Weitere Informationen finden Sie unter login-customer-id.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.

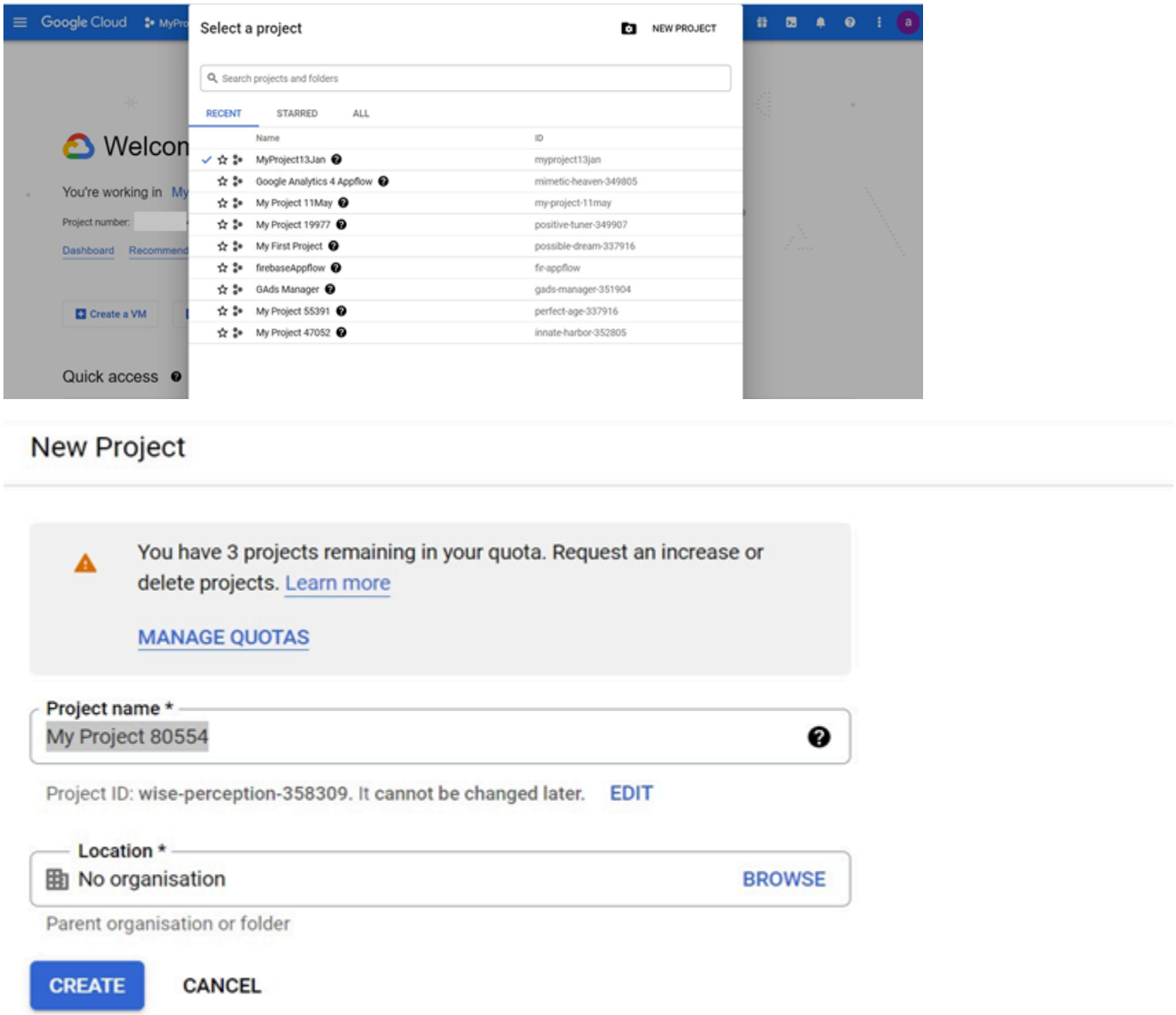
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.

Ein Google Ads-Konto erstellen

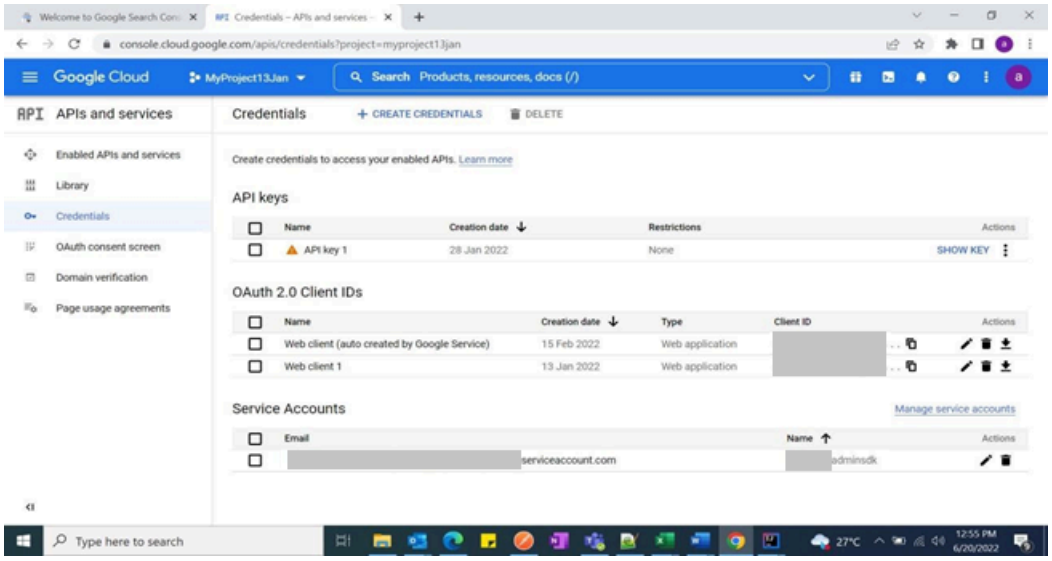
1. Melden Sie sich mit Ihren Anmeldeinformationen beim Google Ads-Entwicklerkonto an und gehen Sie zu *MyProject.



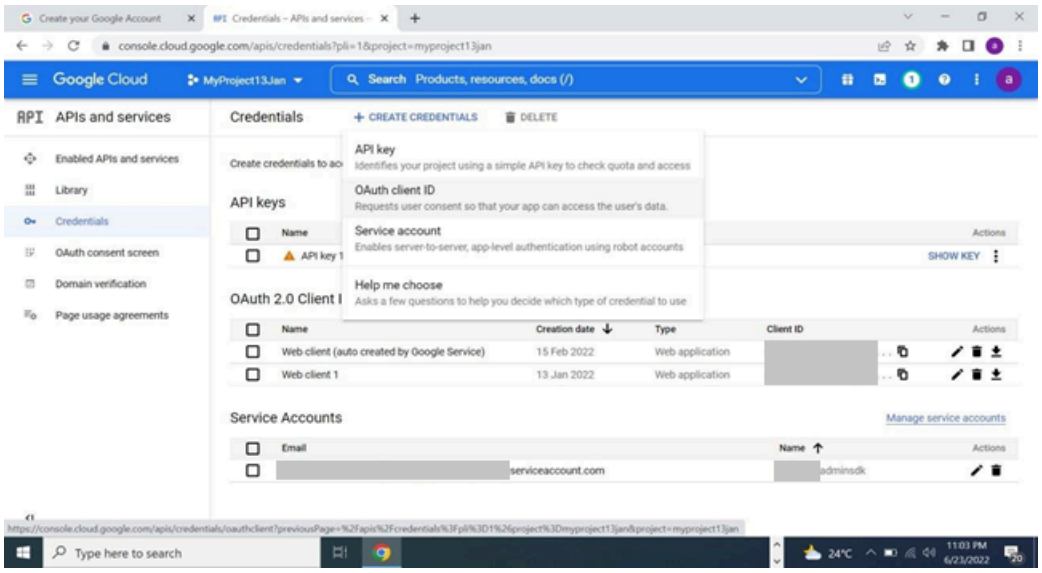
2. Wählen Sie Neues Projekt und geben Sie die Informationen ein, die für die Erstellung eines Google-Projekts erforderlich sind, wenn Sie keine registrierte Anwendung darin haben.



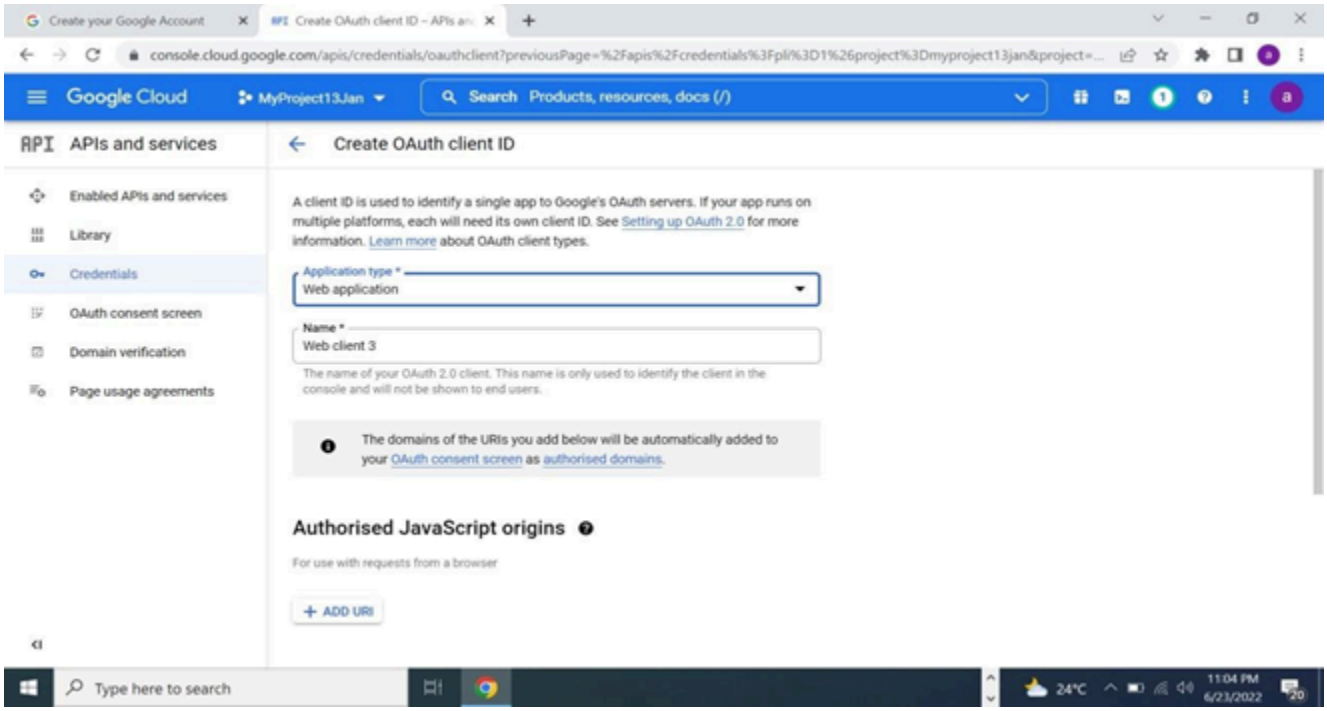
3. Wählen Sie die Registerkarte „Navigation“, dann „API und Einstellungen“ und dann „Client-ID erstellen“. Für diese Option ist eine weitere Konfiguration erforderlich, um eine Verbindung zwischen AWS Glue und herzustellen GoogleAds. ClientSecret Weitere Informationen finden Sie unter API-Anmeldeinformationen.



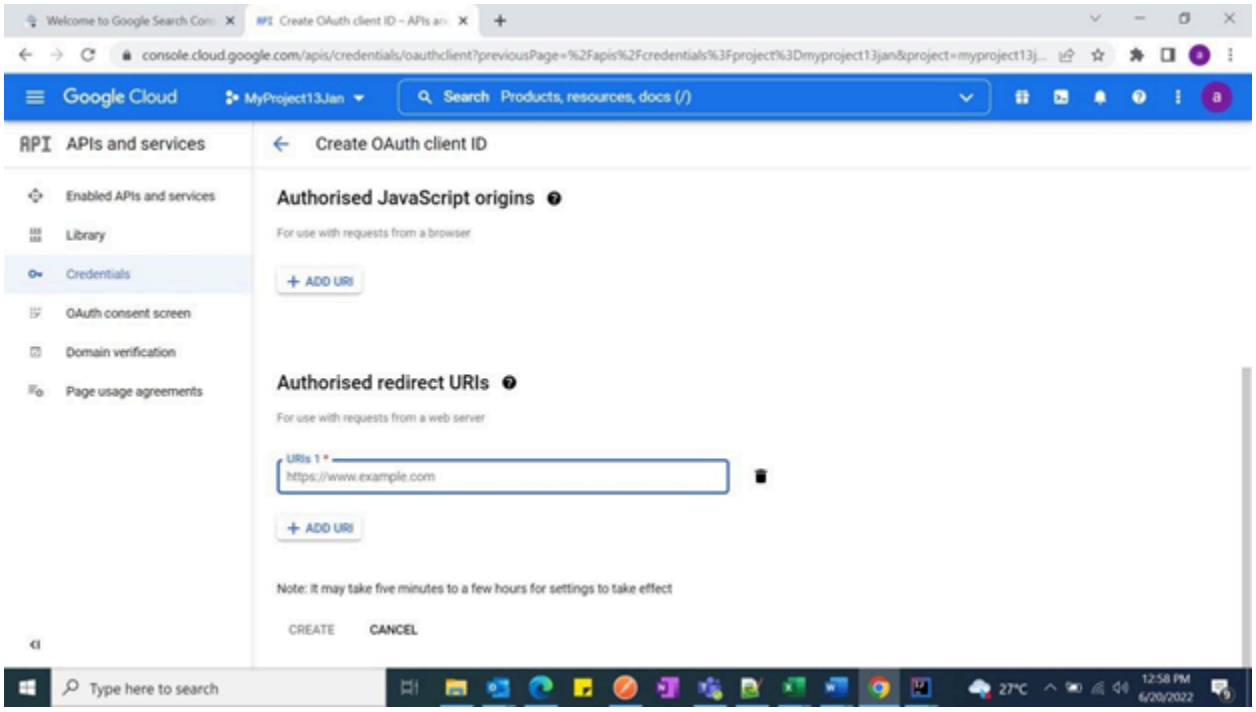
4. Wählen Sie CREATE CREDENTIALS und wählen Sie OAuth Client-ID.



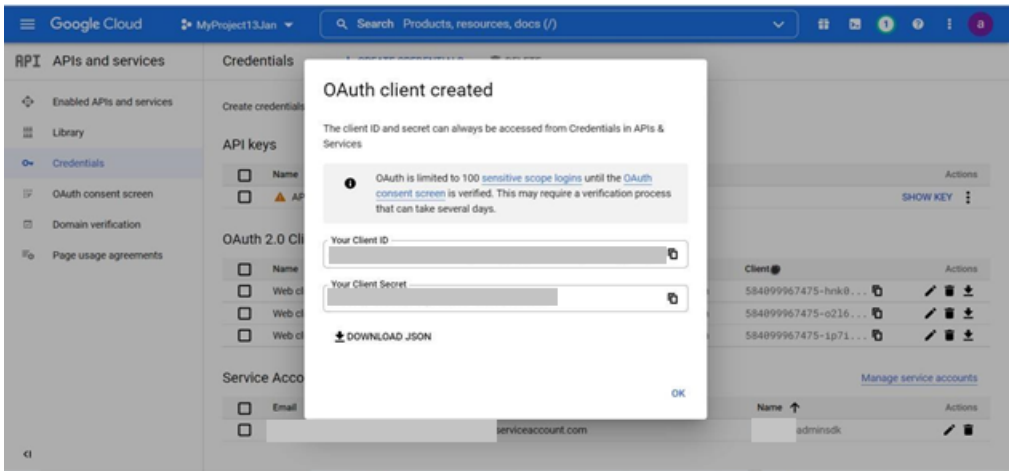
5. Wählen Sie als Anwendungstyp Webanwendung aus.



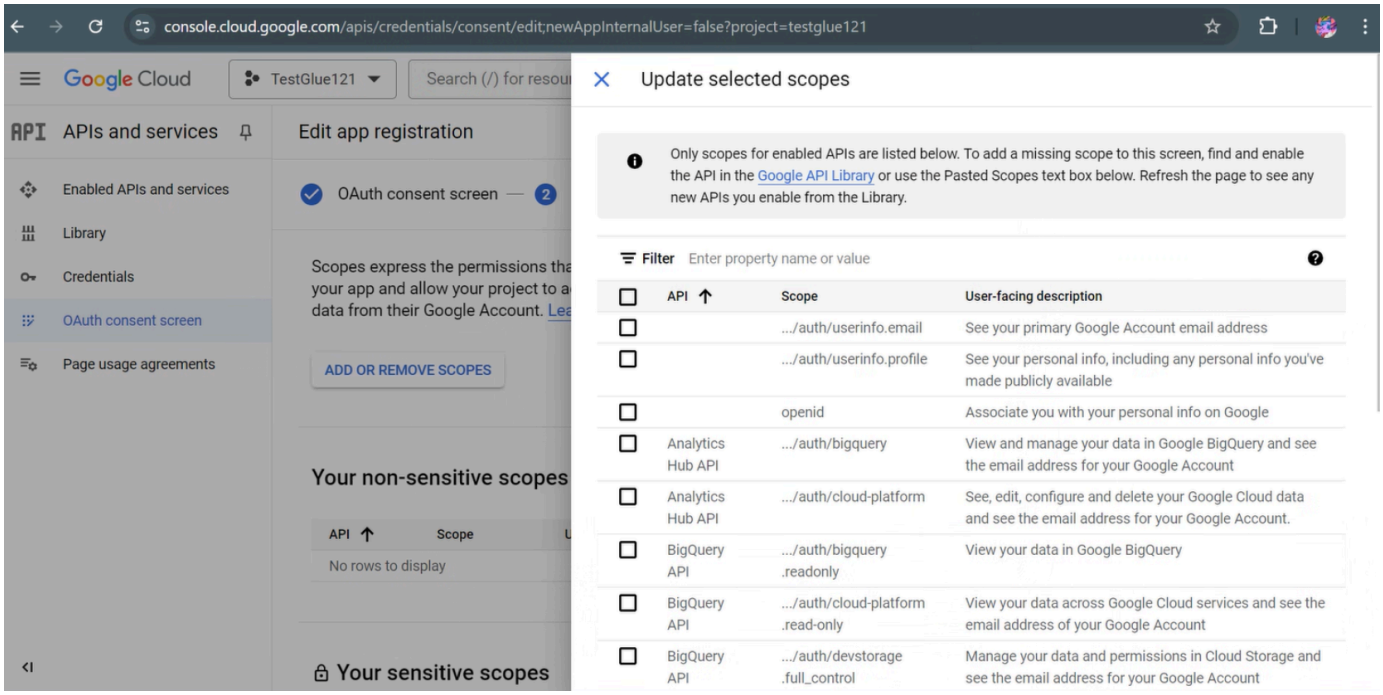
6. Fügen Sie unter Autorisierte Weiterleitung URIs die OAuth Weiterleitung hinzu URIs und wählen Sie Erstellen aus. Sie können bei URIs Bedarf mehrere Weiterleitungen hinzufügen.



7. Ihre Kunden-ID und Ihr geheimer Kundenschlüssel werden generiert, wenn Sie eine Verbindung zwischen AWS Glue und Google Ads herstellen.



8. Fügen Sie die Bereiche entsprechend Ihren Anforderungen hinzu, wählen Sie den OAuth Zustimmungsbildschirm aus, geben Sie die erforderlichen Informationen ein und fügen Sie die Bereiche je nach Anforderung hinzu.



Einschränkungen

Die folgenden Einschränkungen gelten für den Google Ads-Connector:

- `MANAGER_ID`ist eine optionale Eingabe beim Herstellen einer Verbindung. Wenn Sie jedoch auf die Kunden zugreifen möchten, die einem bestimmten Manager zugrunde liegen, `MANAGER_ID` ist dies eine obligatorische Eingabe. In der folgenden Tabelle werden die Zugriffsbeschränkungen je nachdem, ob eine Verbindung enthalten `MANAGER_ID` ist oder nicht, erläutert.

MANAGER_ID wurde beim Herstellen der Verbindung angegeben?	Zugängliche Kunden
Ja	Die Kunden, die unter der angegeben en MANAGER_ID zusammen mit der MANAGER_ID aufgeführt sind.
Nein	Es werden alle Kunden aufgeführt, aber auf Kunden, die einem Manager zugrunde liegen, kann nicht zugegriffen werden.

- Wenn ein Verwaltungskonto als Objekt ausgewählt wird, `Account` wird es nur als Unterobjekt angezeigt. Im Google Ads-Connector werden Entitäten wie Kampagnen, Anzeigen usw. anhand einzelner Kundenkonten abgerufen, nicht anhand des Verwaltungskontos.
- Sie können keine Messwerte für das Verwaltungskonto abrufen. Sie können stattdessen Metriken für einzelne Kundenkonten abrufen.
- Jedes Konto kann bis zu 10.000 Kampagnen enthalten, darunter sowohl aktive als auch pausierte Kampagnen. Weitere Informationen finden Sie unter Kampagne pro Konto.
- Wenn Sie bei der Erstellung eines Berichts bestimmte Metriken für die Anzeige auswählen, werden Zeilen, deren ausgewählte Metriken alle Null sind, nicht zurückgegeben. Weitere Informationen finden Sie unter Zero Metrics.
- Bei den folgenden Feldern funktioniert der Ablauf „Vollständige Zuordnung“ nicht für die Anzeigenentitäten „Account“, „Anzeigengruppe“ und „Anzeigengruppe“, insbesondere für ConversionAction,, conversionActionCategory. conversionActionName Weitere Informationen finden Sie unter Segment und Metriken.
- Ein Datumsbereichsfilter ist erforderlich, wenn das `segments.date` Feld ausgewählt ist.

Verbindung zu Google Analytics herstellen 4

Google Analytics 4 ist ein Analysedienst, der Messwerte über Besucherinteraktionen mit Ihren Apps und Websites verfolgt und meldet. Zu diesen Metriken gehören Seitenaufrufe, aktive Nutzer und Ereignisse. Wenn Sie ein Google Analytics 4-Nutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Google Analytics 4-Konto herstellen. Sie können Google Analytics 4 als Datenquelle in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von Google Analytics 4 an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Google Analytics 4
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Google Analytics 4
- Konfiguration von Google Analytics 4-Verbindungen
- Lesen aus Google Analytics 4-Entitäten
- Verbindungsoptionen für Google Analytics 4
- Ein Google Analytics 4-Konto erstellen
- Schritte zum Erstellen einer Client-App und OAuth 2.0-Anmeldeinformationen
- Einschränkungen und Überlegungen

AWS Glue Unterstützung für Google Analytics 4

AWS Glue unterstützt Google Analytics 4 wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Google Analytics 4 abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Google Analytics 4-API-Versionen

Version 1 Beta.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Google Analytics 4

Bevor Sie die Übertragung von Google Analytics 4 verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Google Analytics-Konto mit einem oder mehreren Datenströmen, die die Daten sammeln, die Sie übertragen möchten.
- Sie haben ein Google Cloud Platform-Konto und ein Google Cloud-Projekt.
- In Ihrem Google Cloud-Projekt haben Sie Folgendes aktiviert APIs:
  - Google Analytics-API

  - Google Analytics-Administrator-API
  - Google Analytics-Daten-API
- In Ihrem Google Cloud-Projekt haben Sie einen OAuth Zustimmungsbildschirm für externe Nutzer konfiguriert. Informationen zum OAuth Einwilligungsbildschirm finden Sie in der Google Cloud Platform Console-Hilfe unter Ihren OAuth Einwilligungsbildschirm einrichten.
- In Ihrem Google Cloud-Projekt haben Sie eine OAuth 2.0-Client-ID konfiguriert. Weitere Informationen finden Sie unter Einrichtung von OAuth 2.0.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Google Analytics 4-Konto herstellen.

Konfiguration von Google Analytics 4-Verbindungen

So konfigurieren Sie eine Google Sheet-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für den AuthorizationCode Zuschusstyp:
      - Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Google Analytics 4.
   b. Geben Sie das `INSTANCE_URL` von Google Analytics 4 an, mit dem Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
```

```
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

`AUTHORIZATION_CODE`Art des Zuschusses.

Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Google Analytics 4 weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Google Analytics 4-Instanz gewähren muss.

Nutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Google Analytics 4 zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Google Analytics 4 weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.

Weitere Informationen finden Sie unter Verwenden von Auth 2.0 für den Zugriff auf Google. APIs

Lesen aus Google Analytics 4-Entitäten

Voraussetzungen

- Ein Google Analytics 4-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Bericht in Echtzeit	Ja	Ja	Ja	Ja	Nein
Kernbericht	Ja	Ja	Ja	Ja	Ja

Beispiel

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta"
    }
```

Entitäts- und Felddetails in Google Analytics 4

Entität	Feld	Datentyp	Unterstützte Operatoren
Kernbericht	Dynamische Felder		
Kernbericht	Dimensionsfelder	String	WIE, =
Kernbericht	Dimensionsfelder	Datum	WIE, =
Kernbericht	Metrische Felder	String	>, <, >=, <=, = ZWISCHEN

Entität	Feld	Datentyp	Unterstützte Operatoren
Kernbericht	Felder für benutzerdefinierte Dimensionen und benutzerdefinierte Metriken	String	N/A
Bericht in Echtzeit	Version der App	String	WIE, =
Bericht in Echtzeit	Zielgruppen-ID	String	WIE, =
Bericht in Echtzeit	Name der Zielgruppe	String	WIE, =
Bericht in Echtzeit	city	String	WIE, =
Bericht in Echtzeit	CityID	String	WIE, =
Bericht in Echtzeit	country	String	WIE, =
Bericht in Echtzeit	CountryID	String	WIE, =
Bericht in Echtzeit	Gerätekategorie	String	WIE, =
Bericht in Echtzeit	eventName	String	WIE, =
Bericht in Echtzeit	Vor Minuten	String	WIE, =
Bericht in Echtzeit	platform	String	WIE, =
Bericht in Echtzeit	streamId	String	WIE, =
Bericht in Echtzeit	streamName	String	WIE, =
Bericht in Echtzeit	unifiedScreenName	String	WIE, =
Bericht in Echtzeit	Aktive Benutzer	String	>, <, >=, <=, = ZWISCHEN
Bericht in Echtzeit	Konvertierungen	String	>, <, >=, <=, = ZWISCHEN

Entität	Feld	Datentyp	Unterstützte Operatoren
Bericht in Echtzeit	Anzahl der Ereignisse	String	>, <, >=, <=, = ZWISCHEN
Bericht in Echtzeit	screenPageViews	String	>, <, >=, <=, = ZWISCHEN

Abfragen partitionieren

1. Filterbasierte Partition

   Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

   - `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
   - `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

     Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:`"2024-02-06"`.
   - `UPPER_BOUND`: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
   - `NUM_PARTITIONS`: Anzahl der Partitionen.

   Beispiel

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta",
        "PARTITION_FIELD": "date"
        "LOWER_BOUND": "2022-01-01"
        "UPPER_BOUND": "2024-01-02"
        "NUM_PARTITIONS": "10"
```

```
    }
```

2. Auf Datensätzen basierende Partition

   Zusätzliche Spark-Optionen NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

   - NUM_PARTITIONS: Anzahl der Partitionen.

   Beispiel

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta",
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Google Analytics 4

Im Folgenden sind die Verbindungsoptionen für Google Analytics 4 aufgeführt:

- ENTITY_NAME(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Google Analytics 4.
- API_VERSION(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Rest-API-Version von Google Analytics 4, die Sie verwenden möchten.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.

- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `INSTANCE_URL`(Integer) — Wird zum Lesen verwendet. (Optional)

Ein Google Analytics 4-Konto erstellen

Folgen Sie den Schritten, um ein Google Analytics 4-Konto zu erstellen: https://support.google.com/analytics/answer/9304153? hl=de

Schritte zum Erstellen einer Client-App und OAuth 2.0-Anmeldeinformationen

Weitere Informationen finden Sie in der Google Analytics4-API-Dokumentation.

1. Erstellen und richten Sie Ihr Konto ein, indem Sie sich mit Ihren Anmeldeinformationen bei Ihrem Google Analytics-Konto anmelden. Navigieren Sie dann zu Admin > Konto erstellen.
2. Erstellen Sie eine Immobilie für das Konto, das Sie erstellt haben, indem Sie „Immobilie erstellen“ wählen. Richten Sie die Immobilie mit den erforderlichen Details ein. Sobald alle Angaben gemacht wurden, wird die entsprechende Immobilien-ID generiert.
3. Fügen Sie einen Datenstream für die erstellte Eigenschaft hinzu, indem Sie im Drop-down-Menü Datenstreams > Stream hinzufügen > Web wählen. Geben Sie die Website-Details wie URL und andere Pflichtfelder an. Nach Angabe aller Details werden die entsprechende Stream-ID und die Mess-ID generiert.
4. Richten Sie Google Analytics auf Ihrer Website ein, indem Sie die Mess-ID kopieren und zur Konfiguration Ihrer Website hinzufügen.
5. Erstellen Sie einen Bericht aus Google Analytics, indem Sie zu Berichte navigieren und den erforderlichen Bericht generieren.
6. Autorisieren Sie Ihre App, indem Sie zu console.cloud.google.com navigieren, nach der Google Analytics-Daten-API suchen und dann die API aktivieren.

    1. Navigieren Sie zur Seite „API und Dienste“ und wählen Sie „Anmeldeinformationen“ > „2.0-Client einrichten“. OAuth IDs

2. Geben Sie die Weiterleitungs-URL an, indem Sie die AWS Glue Weiterleitungs-URL hinzufügen.

7. Kopieren Sie die Client-ID und das Client-Geheimnis, für die Herstellung der Verbindung weitere Informationen erforderlich sind.

Einschränkungen und Überlegungen

Die folgenden Einschränkungen gelten für den Google Analytics 4-Connector:

- Für die Core Report-Entität dürfen nur 9 Dimensionsfelder und 10 Metrikfelder in einer Anfrage gesendet werden. Wenn die zulässige Anzahl von Feldern überschritten wird, schlägt die Anfrage fehl und der Connector gibt eine Fehlermeldung aus.
- Für die Entität Realtime Report dürfen nur 4 Dimensionsfelder eine Anfrage senden. Wenn die zulässige Anzahl von Feldern überschritten wird, schlägt die Anfrage fehl und der Connector gibt eine Fehlermeldung aus.
- Google Analytics 4 ist ein kostenloses Tool in der Betaversion. Daher wird es regelmäßig Updates zu neuen Funktionen, zur Verbesserung von Entitäten, zum Hinzufügen neuer Felder und zur Ablehnung vorhandener Felder geben.
- Die Felder des Kernberichts werden dynamisch gefüllt, sodass Felder hinzugefügt, abgewertet und umbenannt werden können. Außerdem können jederzeit neue Beschränkungen für Felder festgelegt werden.
- Das Standardstartdatum ist 30 Tage und das Enddatum ist gestern (ein Tag vor dem aktuellen Datum). Diese Daten werden im Filterausdruckscode überschrieben, wenn der Benutzer den Wert festgelegt hat ODER wenn der Ablauf inkrementell ist.
- Gemäß der Dokumentation gibt die Echtzeit-Berichtseinheit 10.000 Datensätze zurück, wenn das Limit in der Anfrage nicht überschritten wird. Andernfalls gibt die API maximal 250.000 Zeilen pro Anfrage zurück, unabhängig davon, nach wie vielen Sie fragen. Weitere Informationen finden Sie unter Methode: Eigenschaften. runRealtimeReportin der Google Analytics-Dokumentation.
- Die Entität für Echtzeitberichte unterstützt keine datensatzbasierte Partition, da sie keine Paginierung unterstützt. Außerdem unterstützt sie keine feldbasierte Partition, da keines der Felder die definierten Kriterien erfüllt.
- Aufgrund der Beschränkung der Anzahl der Felder, die in einer Anfrage übergeben werden können. Wir legen die Standardfelder für Dimensionen und Metriken innerhalb der festgelegten Grenzwerte fest. Wenn „Alles auswählen“ ausgewählt ist, werden nur die Daten aus diesen vordefinierten Feldern abgerufen.

- Kernbericht
  - Gemäß der SAAS-Beschränkung sind Anfragen mit bis zu 9 Dimensionen und nur bis zu 10 Metriken zulässig (d. h., eine Anfrage kann maximal 19 Felder enthalten (Metriken + Dimension).
  - Gemäß der Implementierung: Wenn der Benutzer SELECT_ALL oder mehr als 25 ausgewählte Felder verwendet, werden Standardfelder in der Anfrage übergeben.
  - Die folgenden Felder werden als Standardfelder für Core Report betrachtet: „country“, „city“, „eventName“, „cityID“, „browser“, „date“, „currencyCode“, „deviceCategory“, „transactionId“, active1 „, „active28 „, „active7 DayUsers „, DayUsers „activeUsers“, "," User“, "DayUsers„, „EngagedSessions“, „EventCount“, averagePurchaseRevenue „Engagement-Rate“. averageRevenuePer averageSessionDuration
- Bericht in Echtzeit
  - Gemäß den Einschränkungen von SAAS-Anfragen sind bis zu 4 Dimensionen zulässig.
  - Wenn der Benutzer SELECT_ALL oder mehr als 15 Felder ausgewählt hat, werden Standardfelder in der Anfrage übergeben.
  - Die folgenden Felder werden als Standardfelder für den RealTime Bericht betrachtet: „Land“, „Gerätekategorie“, „Stadt“, „CityID“, „ActiveUsers“, „Konversionen“, „EventCount“, "“. screenPageViews
- In der Core-Report-Entität, wenn Partition im Datumsfeld und Filter auf StartDate gleichzeitig vorhanden sind. In diesem Fall wird der DateRange-Wert mit dem StartDate-Filterwert überschrieben, aber da die Partition immer die Priorität haben muss, wird der StartDate-Filter verworfen, wenn das Feld Partition auf Datum bereits vorhanden ist.
- Da CohortSpecs jetzt auch Teil der Core-Report-Anfrage ist, haben wir die aktuelle Core-Report-Entität um Unterstützung für das CohortSpec-Attribut erweitert. Im CohortSpecs-Anfragetext erfordern fast alle Felder Benutzereingaben. Um dieses Problem zu lösen, haben wir Standardwerte für diese Attribute/Felder festgelegt und dem Benutzer die Möglichkeit gegeben, diese Werte bei Bedarf zu überschreiben.

FieldName	Standardwerte	Beispielabfrage zur Übergabe von filterPredicate-Optionen zum Überschreiben von Standardwerten
startDate	vor 30 Tagen ab dem aktuellen Datum	„StartDate zwischen „2023-05-09" und „2023-05-10"
endDate	vor einem Tag ab dem aktuellen Datum	„StartDate zwischen „2023-05-09" und „2023-05-10"
startOffset	0	StartOffset=2
EndOffset	1	EndOffset=10
Granularität	TÄGLICH	Granularity="Wöchentlich“

- Sie können all diese Filter auch zusammen oder zusammen mit anderen Filtern verwenden.
  - Beispiel 1 — FilterPredicate: StartDate zwischen „2023-05-09" und „2023-05-10" UND startOffset=1 UND endOffset=2 UND granularity="Weekly“
  - Beispiel 2 — FilterPredicate: city= „xyz“ UND startOffset=1 UND endOffset=2 UND granularity="Weekly“
- In einer Kohortenanfrage:
  - Wenn 'cohortNthMonth' in der Anfrage übergeben wird, wird der interne Granularitätswert auf „MONTHLY“ gesetzt
  - In ähnlicher Weise wird, wenn cohortNthWeek '' übergeben wird, der Granularitätswert auf „WÖCHENTLICH“ gesetzt
  - Und für 'cohortNthDay' wird der Granularitätswert auf „TÄGLICH“ gesetzt. Weitere Informationen finden Sie unter:
    - https://developers.google.com/analytics/devguides/reporting/data/v1/advanced
    - https://developers.google.com/analytics/devguides/reporting/data/v1/rest/v1beta/CohortSpec
  - Es ist vorgesehen, dass der Benutzer den Standardwert für DateRange und Granularität überschreibt. Weitere Informationen finden Sie in der obigen Tabelle.

Verbindung zu Google herstellen BigQuery in AWS Glue Studio

> **Note**
>
> Sie können Folgendes verwenden … AWS Glue für Spark zum Lesen und Schreiben in Tabellen in Google BigQuery in AWS Glue 4.0 und spätere Versionen. Um Google zu konfigurieren BigQuery mit AWS Glue Jobs programmgesteuert, siehe BigQuery Verbindungen.

AWS Glue Studio bietet eine visuelle Oberfläche, mit der Sie eine Verbindung herstellen BigQuery, Datenintegrationsjobs erstellen und diese ausführen können AWS Glue Studio Serverlose Spark-Laufzeit.

Beim Herstellen einer Verbindung zu Google BigQuery in AWS Glue Studio wird eine einheitliche Verbindung hergestellt. Weitere Informationen finden Sie unter Überlegungen.

Anstatt ein Geheimnis mit den Anmeldeinformationen in einem bestimmten Format zu erstellen`{"credentials": "base64 encoded JSON"}`, können Sie jetzt mit einer einheitlichen Verbindung zu Google ein Geheimnis erstellen BigQuery, das direkt den JSON von Goolge BigQuery enthält:. `{"type": "service-account", ...}`

Themen

- BigQuery Verbindung herstellen
- Einen BigQuery Quellknoten erstellen
- Einen BigQuery Zielknoten erstellen
- Erweiterte Optionen

BigQuery Verbindung herstellen

Um BigQuery von aus eine Verbindung zu Google herzustellen AWS Glue, müssen Sie Ihre Anmeldeinformationen für die Google Cloud Platform erstellen und AWS Secrets Manager geheim speichern und dieses Geheimnis dann einer BigQuery AWS Glue Google-Verbindung zuordnen.

So konfigurieren Sie eine Verbindung zu BigQuery:

1. Erstellen und identifizieren Sie in der Google Cloud Platform relevante Ressourcen:

- Erstellen oder identifizieren Sie ein GCP-Projekt mit BigQuery Tabellen, zu denen Sie eine Verbindung herstellen möchten.
- Aktivieren Sie die BigQuery API. Weitere Informationen finden Sie unter Verwenden der BigQuery Storage Read API zum Lesen von Tabellendaten.

2. Erstellen und exportieren Sie in Google Cloud Platform Anmeldeinformationen für ein Dienstkonto:

   Sie können den Assistenten für BigQuery Anmeldeinformationen verwenden, um diesen Schritt zu beschleunigen: Anmeldeinformationen erstellen.

   Folgen Sie der Anleitung unter Dienstkonten erstellen, um ein Dienstkonto in GCP zu erstellen.

   - Wählen Sie bei der Projektauswahl das Projekt aus, das Ihre BigQuery Tabelle enthält.
   - Wenn Sie GCP-IAM-Rollen für Ihr Dienstkonto auswählen, fügen Sie eine Rolle hinzu oder erstellen Sie eine Rolle, die entsprechende Berechtigungen zum Ausführen von BigQuery Jobs zum Lesen, Schreiben oder Erstellen BigQuery von Tabellen gewährt.

   Folgen Sie der Anleitung unter Einen Dienstkontoschlüssel erstellen, um Anmeldeinformationen für Ihr Dienstkonto zu erstellen.

   - Wählen Sie für den Schlüsseltyp JSON aus.

   Sie sollten jetzt eine JSON-Datei mit Anmeldeinformationen für Ihr Dienstkonto heruntergeladen haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
}
```

3. Erstellen Sie AWS Secrets Manager unter mithilfe Ihrer heruntergeladenen Anmeldeinformationsdatei ein Geheimnis. Sie können die Registerkarte Klartext wählen und den Inhalt der Datei im JSON-Format einfügen. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.
4. Stellen Sie im AWS Glue Datenkatalog eine Verbindung her, indem Sie den Schritten unter folgenhttps://docs.aws.amazon.com/glue/latest/dg/console-connections.html. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *connectionName*
   - Wählen Sie bei der Auswahl eines Verbindungstyps Google aus BigQuery.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*secretName*.
5. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle *secretName* Leserechte.
6. Stellen *connectionName* Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Einen BigQuery Quellknoten erstellen

Voraussetzungen

- Eine Verbindung BigQuery vom Typ AWS Glue Datenkatalog
- Ein AWS Secrets Manager Geheimnis für Ihre BigQuery Google-Anmeldeinformationen, das von der Verbindung verwendet wird.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Der Name und der Datensatz der Tabelle und des entsprechenden Google Cloud-Projekts, das gelesen werden soll.

Eine BigQuery Datenquelle hinzufügen

Um einen BigQueryDatenquellen-Node hinzuzufügen:

1. Wählen Sie die Verbindung für Ihre BigQuery Datenquelle. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung herstellen müssen, wählen

Sie BigQuery Verbindung erstellen. Weitere Informationen finden Sie unter Übersicht über die Verwendung von Konnektoren und Verbindungen.

Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.

2. Identifizieren Sie, welche BigQuery Daten Sie lesen möchten, und wählen Sie dann eine BigQuery Quelloption

   - Einzelne Tabelle auswählen – ermöglicht es Ihnen, alle Daten aus einer Tabelle abzurufen.
   - Benutzerdefinierte Abfrage eingeben – ermöglicht Ihnen, durch Angabe einer Abfrage anzupassen, welche Daten abgerufen werden.

3. Beschreiben Sie die Daten, die gelesen werden sollen

   (Erforderlich) Geben Sie als Übergeordnetes Projekt das Projekt, das Ihre Tabelle enthält, oder, falls relevant, ein übergeordnetes Fakturierungsprojekt an.

   Wenn du eine einzelne Tabelle ausgewählt hast, setze Table auf den Namen einer BigQuery Google-Tabelle im folgenden Format: `[dataset].[table]`

   Wenn Sie eine Abfrage ausgewählt haben, tun Sie dies für Abfrage. Verweisen Sie in Ihrer Abfrage mit ihrem vollständig qualifizierten Tabellennamen im folgenden Format auf Tabellen: `[project].[dataset].[tableName]`.

4. Geben Sie BigQuery Eigenschaften an

   Wenn Sie eine einzelne Tabelle ausgewählt haben, müssen Sie keine zusätzlichen Eigenschaften angeben.

   Wenn Sie eine Abfrage ausgewählt haben, müssen Sie die folgenden benutzerdefinierten BigQuery Google-Eigenschaften angeben:

   - Setzen Sie `viewsEnabled` auf „true“.
   - Legen Sie `materializationDataset` auf einen Datensatz fest. Der GCP-Prinzipal, der anhand der über die AWS Glue Verbindung bereitgestellten Anmeldeinformationen authentifiziert wurde, muss in der Lage sein, Tabellen in diesem Datensatz zu erstellen.

Einen BigQuery Zielknoten erstellen

Voraussetzungen

- Eine Verbindung BigQuery vom Typ AWS Glue Datenkatalog
- Ein AWS Secrets Manager Geheimnis für Ihre BigQuery Google-Anmeldeinformationen, das von der Verbindung verwendet wird.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Der Name und der Datensatz der Tabelle und des entsprechenden Google Cloud-Projekts, in die geschrieben werden soll.

Ein BigQuery Datenziel hinzufügen

Um einen BigQueryDatenziel-Knoten hinzuzufügen:

1. Wählen Sie die Verbindung für Ihr BigQuery Datenziel. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung herstellen müssen, wählen Sie BigQuery Verbindung erstellen. Weitere Informationen finden Sie unter Übersicht über die Verwendung von Konnektoren und Verbindungen.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Identifizieren Sie, BigQuery in welche Tabelle Sie schreiben möchten, und wählen Sie dann eine Schreibmethode aus.

   - Direkt — schreibt BigQuery direkt über die BigQuery Storage Write-API in.
   - Indirekt — schreibt in Google Cloud Storage und kopiert dann nach BigQuery.

   Wenn Sie indirekt schreiben möchten, geben Sie einen GCS-Zielort mit einem temporären GCS-Bucket an. Sie müssen zusätzliche Konfigurationen für Ihre AWS Glue Verbindung angeben. Weitere Informationen finden Sie unter Indirektes Schreiben mit Google verwenden BigQuery.
3. Beschreiben Sie die Daten, die gelesen werden sollen

   (Erforderlich) Geben Sie als Übergeordnetes Projekt das Projekt, das Ihre Tabelle enthält, oder, falls relevant, ein übergeordnetes Fakturierungsprojekt an.

Wenn Sie eine einzelne Tabelle ausgewählt haben, setzen Sie Table auf den Namen einer BigQuery Google-Tabelle im folgenden Format: `[dataset].[table]`

Erweiterte Optionen

Sie können beim Erstellen eines Knotens erweiterte Optionen angeben. BigQuery Diese Optionen sind dieselben wie bei der Programmierung AWS Glue für Spark-Skripte.

Weitere Informationen finden Sie in der Referenz zu den BigQuery Verbindungsoptionen im AWS Glue Entwicklerhandbuch.

Verbindung zur Google Search Console herstellen

Die Google Search Console ist eine kostenlose Plattform, die Website-Besitzern zur Verfügung steht, um zu überwachen, wie Google die Website betrachtet, und um ihre organische Präsenz zu optimieren. Dazu gehören die Anzeige verweisender Domains, die Leistung mobiler Websites, umfangreiche Suchergebnisse sowie die Anfragen und Seiten mit dem höchsten Traffic. Wenn Sie ein Nutzer der Google Search Console sind, können Sie eine Verbindung AWS Glue zu Ihrem Google Search Console-Konto herstellen. Sie können die Google Search Console als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von der Google Search Console an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für die Google Search Console
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration der Google Search Console
- Google Search Console-Verbindungen konfigurieren
- Aus Google Search Console-Entitäten lesen
- Verbindungsoptionen für die Google Search Console
- Einschränkungen der Google Search Console

AWS Glue Unterstützung für die Google Search Console

AWS Glue unterstützt die Google Search Console wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus der Google Search Console abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen der Google Search Console

Die folgenden API-Versionen der Google Search Console werden unterstützt:

- v3

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration der Google Search Console

Bevor Sie Daten aus der Google Search Console übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Google Search Console-Konto.
- Sie haben ein Google Cloud Platform-Konto und ein Google Cloud-Projekt.
- In Ihrem Google Cloud-Projekt haben Sie die Google Search Console-API aktiviert.
- In Ihrem Google Cloud-Projekt haben Sie einen OAuth Zustimmungsbildschirm für externe Nutzer konfiguriert. Weitere Informationen finden Sie in der Google Cloud Platform Console-Hilfe unter Ihren OAuth Einwilligungsbildschirm einrichten.
- In Ihrem Google Cloud-Projekt haben Sie eine OAuth 2.0-Client-ID konfiguriert. Informationen zu den Kundenanmeldedaten, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden, finden Sie unter Einrichtung von OAuth 2.0.

Wenn du diese Anforderungen erfüllst, bist du bereit, eine Verbindung AWS Glue zu deinem Google Search Console-Konto herzustellen. Für typische Verbindungen müssen Sie in der Google Search Console nichts weiter tun.

Google Search Console-Verbindungen konfigurieren

Die Google Search Console unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2 Die Art der Gewährung bestimmt, wie mit der Google Search Console AWS Glue kommuniziert wird, um Zugriff auf Ihre Daten anzufordern.

- Diese Art der Gewährung gilt als „dreibeiniges“, OAuth da sie darauf beruht, dass Nutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Nutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in der Google Search Console zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zur Google Search Console weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
- Eine öffentliche Dokumentation der Google Search Console zur Erstellung einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter Verwenden von OAuth 2.0 für den Zugriff auf Google APIs.

So konfigurieren Sie eine Verbindung zur Google Search Console:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps die Google Search Console aus.

b. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

c. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

d. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Google Search Console-Entitäten lesen

Voraussetzung

Ein Objekt der Google Search Console, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Suchanalytik	Ja	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Standorte	Nein	Nein	Nein	Ja	Nein
Sitemaps	Nein	Nein	Nein	Ja	Nein

Beispiel:

```
googleSearchConsole_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesearchconsole",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v3"
    }
```

Entitäts- und Felddetails der Google Search Console:

Die Google Search Console bietet Endpunkte zum dynamischen Abrufen von Metadaten für unterstützte Entitäten. Dementsprechend wird die Operatorunterstützung auf der Datentypebene erfasst.

Entität	Feld	Datentyp	Unterstützte Operatoren	Hinweis
Suchen Sie nach Analytik	keys	Auflisten	N/A	
	Klicks	Double	N/A	
	Impressionen	Double	N/A	
	ctr	BigDecimal	N/A	Für den BigDecimal Datentyp wird der Wert '0' als '0E-18' formatiert

Entität	Feld	Datentyp	Unterstützte Operatoren	Hinweis
	position	Double	N/A	
	start_end_date	Datum	BETWEEN	<30 days ago from the current date>Der Standardwert für liegt zwischen UND `start_end_date` <yesterday: that is, 1 day ago from the current date> Hinweis: Erwartet, dass Sie einen UTC-Datumswert übergeben. Beispiel: start_end_date zwischen '2021-01-01' UND '2024-09-09'
	country	String	EQUAL_TO, NOT_EQUAL_TO, ENTHÄLT	Gültige Werte sind „IND“, „CAN“ usw.

Entität	Feld	Datentyp	Unterstützte Operatoren	Hinweis
	Typ	String	EQUAL_TO, NOT_EQUAL_TO	Gültige Werte sind „discover“, „GoogleNews“, „news“, „image“, „video“, „web“
	Aussehen der Suche	String	EQUAL_TO, NOT_EQUAL_TO, ENTHÄLT	Eine Liste gültiger Werte finden Sie unter Darstellung der Suche.
	Gerät	String	EQUAL_TO, NOT_EQUAL_TO, ENTHÄLT	Gültige Werte sind „DESKTOP“, „MOBILE“, „TABLET“
	Dimensionen	String	EQUAL_TO	Gültige Werte sind „Land“, „Gerät“
	angezeigten	String	EQUAL_TO, NOT_EQUAL_TO, ENTHÄLT	
	query	String	EQUAL_TO, NOT_EQUAL_TO, ENTHÄLT	
	Datenstatus	String	EQUAL_TO	Gültige Werte sind „all“ und „final“
Standorte	Site-URL	String	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	Hinweis
	Berechtigungsstufe	String	N/A	
Sitemaps	Pfad	String	N/A	
	Typ	String	N/A	
	Zuletzt eingereicht	DateTime	N/A	
	steht noch aus	Boolesch	N/A	
	isSitemapsIndex	Boolesch	N/A	
	Zuletzt heruntergeladen	DateTime	N/A	
	Warnungen	Long	N/A	
	Fehler	Long	N/A	
	Inhalt	Auflisten	N/A	

> **Note**
>
> Eine aktualisierte Liste gültiger Werte für Filter finden Sie in den API-Dokumenten der Google Search Console.
> Das Feld `start_end_date` ist eine Kombination aus `start_date` und`end_date`.

Abfragen partitionieren

Filterbasierte Partitionierung und datensatzbasierte Partitionierung werden nicht unterstützt.

Verbindungsoptionen für die Google Search Console

Im Folgenden sind die Verbindungsoptionen für die Google Search Console aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in der Google Search Console.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Rest-API-Version der Google Search Console, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`<30 days ago from current date><yesterday: that is, 1 day ago from the current date>(Zeichenfolge) — Standard: „start_end_date zwischen UND“. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: „start_end_date between <30 days ago from current date>AND <yesterday: that is, 1 day ago from the current date>“ wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `INSTANCE_URL`(Zeichenfolge) — Wird zum Lesen verwendet. Eine gültige Google Search Console-Instanz-URL.

Einschränkungen der Google Search Console

Im Folgenden finden Sie Einschränkungen oder Hinweise für die Google Search Console:

- Die Google Search Console erzwingt Nutzungsbeschränkungen für die API. Weitere Informationen finden Sie unter Nutzungsbeschränkungen.
- Wenn kein Filter für die `Search Analytics` Entität übergeben wurde, summiert die API alle Klicks, Impressionen, CTR und andere Daten für Ihre gesamte Website innerhalb des angegebenen Standarddatumsbereichs und präsentiert sie als einen einzigen Datensatz.
- Um die Daten in kleinere Segmente aufzuteilen, müssen Sie Ihrer Abfrage Dimensionen hinzufügen. Dimensions teilt der API mit, wie Sie Ihre Daten segmentieren möchten.
  - Wenn Sie beispielsweise hinzufügen, erhalten `filterPredicate: dimensions="country"` Sie einen Datensatz für jedes Land, in dem Ihre Website im angegebenen Zeitraum Traffic erhalten hat.
  - Beispiel für die Übergabe mehrerer Dimensionen:`filterPredicate: dimensions="country" AND dimensions="device" AND dimensions="page"`. In diesem Fall erhalten Sie für jede eindeutige Kombination dieser drei Dimensionen eine Zeile in der Antwort.
- Für die `dataState` Felder `start_end_date` und sind Standardwerte festgelegt.

Feldname	Standardfilterausdruck	Ausdruck zum Überschreiben von Standardwerten
start_end_date	<30 days ago from current date>start_end_date zwischen UND <yesterday: that is, 1 day ago from the current date>	start_end_date zwischen „2024-01-01" UND „2024-05-05"
Datum/Status	dataState="alles“	dataState="Endgültig“

Verbindung zu Google Sheets herstellen

Google Sheets ist eine Online-Tabellenkalkulationssoftware, mit der Sie große Datenmengen organisieren, benutzerdefinierte Berichte erstellen, Berechnungen automatisieren und mit anderen zusammenarbeiten können. Wenn Sie ein Google Sheets-Nutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Google Sheets-Konto herstellen. Anschließend können Sie Google Sheets als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Google Sheets und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Google Sheets
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Google Sheets konfigurieren
- Google Sheets-Verbindungen konfigurieren
- Aus Google Sheets-Entitäten lesen
- Verbindungsoptionen für Google Sheets
- OAuth Autorisierungscodefluss für Google Sheets einrichten
- Einschränkungen für den Google Sheets-Connector

AWS Glue Unterstützung für Google Sheets

AWS Glue unterstützt Google Sheets wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Google Sheets abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte Google Sheets API-Versionen

Google Sheets API v4 und Google Drive API v3

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Google Sheets konfigurieren

Bevor Sie die Übertragung von Google Sheets verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Google Sheets-Konto mit E-Mail und Passwort.
- Ihr Google Sheets-Konto ist für den API-Zugriff aktiviert. Jegliche Nutzung der Google Sheets API ist ohne zusätzliche Kosten möglich.
- Mit Ihrem Google Sheets-Konto können Sie verbundene Apps installieren. Wenn Sie keinen Zugriff auf diese Funktion haben, wenden Sie sich an Ihren Google Sheets-Administrator.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Google Sheets-Konto herstellen.

Google Sheets-Verbindungen konfigurieren

So konfigurieren Sie eine Google Sheet-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für den AuthorizationCode Zuschusstyp:
      - Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Google Sheets aus.
   b. Stellen Sie die Google Sheets-Umgebung bereit.
      i. Wählen Sie `secretName` die aus, in AWS Glue die Sie für diese Verbindung die Token einfügen möchten.
      ii. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

AUTHORIZATION_CODE Art der Gewährung

Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Nutzer zu Google Sheets weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Google Sheets-Instanz gewähren muss.

Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Google Sheets zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihr

eigenes Client-Geheimnis anzugeben. In diesem Szenario werden sie weiterhin zu Google Sheets weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.

Weitere Informationen finden Sie in der öffentlichen Google Sheets-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss.

Aus Google Sheets-Entitäten lesen

Voraussetzungen

- Ein Google SpreadSheet , aus dem Sie gerne lesen würden. Sie benötigen die SpreadSheet ID und den TabNamen der Tabelle.

Entitäts- und Felddetails von Google Sheets:

Entität	Datentyp	Unterstützte Operatoren
Tabellenkalkulation	String	N/A (Filter wird nicht unterstützt)

Beispiel

```
googleSheets_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesheets",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "{SpreadSheetID}#{SheetTabName}",
        "API_VERSION": "v4"
    }
```

Partitionierung von Abfragen

Nur für die Partitionierung auf Datensatzbasis. `NUM_PARTITIONS` Kann als zusätzliche Spark-Option bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter würde

die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Beispiel mit **NUM_PARTITIONS**

```
googlesheets_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesheets",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "{SpreadSheetID}#{SheetTabName}",
        "API_VERSION": "v4",
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Google Sheets

Im Folgenden sind die Verbindungsoptionen für Google Sheets aufgeführt:

- ENTITY_NAME(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Das SpreadSheet ID und sheetTabName in Google Sheets. Beispiel: {SpreadSheetID}#{SheetTabName}.
- API_VERSION(String) — (Erforderlich) Wird zum Lesen verwendet. Die Rest-API-Version von Google Sheets, die Sie verwenden möchten.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- NUM_PARTITIONS(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

OAuth Autorisierungscodefluss für Google Sheets einrichten

Voraussetzungen

- Ein Google-Konto, in dem Sie sich anmelden können, um die Google Sheets-App zu verwenden. In Ihrem Google-Konto enthält Google Sheets die Daten, die Sie übertragen möchten.

- Ein Google Cloud Platform-Konto und ein Google Cloud-Projekt. Weitere Informationen finden Sie unter Google Cloud-Projekt erstellen.

So richten Sie Ihr Google-Konto ein und erhalten OAuth 2.0-Anmeldeinformationen:

1. Sobald das Google Cloud-Projekt eingerichtet ist, aktivieren Sie die Google Sheets-API und Google Drive APIs im Projekt. Die Schritte zu ihrer Aktivierung finden Sie unter Aktivieren und Deaktivieren APIs in der API-Konsolen-Hilfe für die Google Cloud Platform.
2. Als Nächstes konfigurieren Sie einen OAuth Zustimmungsbildschirm für externe Nutzer. Weitere Informationen zum OAuth Einwilligungsbildschirm finden Sie in der Google Cloud Platform Console-Hilfe unter Ihren OAuth Einwilligungsbildschirm einrichten.
3. Fügen Sie im OAuth Zustimmungsbildschirm die folgenden Bereiche hinzu:

   - Der schreibgeschützte Bereich der Google Sheets-API
   - Der schreibgeschützte Bereich der Google Drive-API

   Weitere Informationen zu diesen Bereichen finden Sie unter OAuth 2.0 Scopes for Google APIs in der Google Identity-Dokumentation.
4. Generieren Sie OAuth 2.0-Client-ID und Secret. Die Schritte zum Erstellen dieser Client-ID finden Sie unter Einrichtung von OAuth 2.0 in der Google Cloud Platform Console-Hilfe.

   Die OAuth 2.0-Client-ID muss über eine oder mehrere autorisierte Weiterleitungen verfügen URLs.

   Weiterleitungen URLs haben das folgende Format:

   - https://<aws-region>.console.aws.amazon. com/gluestudio/oauth
5. Notieren Sie sich die Client-ID und das Client-Geheimnis aus den Einstellungen für Ihre OAuth 2.0-Client-ID.

Einschränkungen für den Google Sheets-Connector

Die folgenden Einschränkungen gelten für den Google Sheets-Connector:

- Der Google Sheets-Connector unterstützt keine Filter. Daher kann eine filterbasierte Partitionierung nicht unterstützt werden.

- Bei der Record Base Partitioning gibt es keine Möglichkeit, die genaue Anzahl von Datensätzen per SAAS zurückzugeben. Daher kann es Szenarien geben, in denen Dateien mit leeren Datensätzen erstellt werden.
- Da der Google Sheets-Connector keine filterbasierte Partitionierung unterstützt, sind `partitionFieldlowerbound`, und keine gültigen Verbindungsoptionen `upperbound` sind. Wenn diese Optionen bereitgestellt werden, ist davon auszugehen, dass der AWS Glue Job fehlschlägt.
- Es ist wichtig, die erste Zeile des Blattes als Kopfzeile festzulegen, um Probleme mit der Datenverarbeitung zu vermeiden.
  - Falls nicht angegeben, wird die Kopfzeile durch,,`Unnamed:3`... ersetzt `Unnamed:1Unnamed:2`, wenn das Blatt Daten enthält, bei denen die erste Zeile leer ist.
  - Wenn die Kopfzeile angegeben wird, werden leere Spaltennamen durch ersetzt`Unnamed:<number of column>`. Wenn die Kopfzeile beispielsweise ist`['ColumnName1', 'ColumnName2', '', '', 'ColumnName5', 'ColumnName6']`, dann wird sie `['ColumnName1', 'ColumnName2', 'Unnamed:3', 'Unnamed:4', 'ColumnName5', 'ColumnName6'].`
- Der Google Sheets-Connector unterstützt keine inkrementelle Übertragung.
- Der Google Sheets-Connector unterstützt nur den Datentyp „Zeichenfolge“.
- Doppelte Kopfzeilen in einem Blatt werden iterativ mit einem numerischen Suffix umbenannt. Vom Benutzer angegebene Kopfzeilennamen haben Vorrang beim Umbenennen der doppelten Kopfzeilen. Wenn die Kopfzeile beispielsweise ["Name“, „“, „Name“, null, „Unnamed:6", „"] ist, ändert sie sich in: ["Name“, „Unnamed:2", „Name1", „Unnamed:4", „Unnamed:4", „Unnamed:6", „Unnamed:61"].
- Der Google Sheets-Connector unterstützt keine Leerzeichen für einen TabNamen.
- Ein Ordnername darf die folgenden Sonderzeichen nicht enthalten:
  - #
  - /

Verbindung herstellen zu HubSpot

HubSpotDie CRM-Plattform bietet alle Tools und Integrationen, die Sie für Marketing, Vertrieb, Inhaltsmanagement und Kundenservice benötigen.

- Marketing Hub — Marketing-Software, mit der Sie den Traffic steigern, mehr Besucher konvertieren und komplette Inbound-Marketingkampagnen in großem Maßstab durchführen können.
- Sales Hub — CRM-Software für den Vertrieb, mit der Sie tiefere Einblicke in potenzielle Kunden erhalten, Ihre Aufgaben automatisieren und mehr Geschäfte schneller abschließen können.
- Service Hub — Kundenservice-Software, die Ihnen hilft, mit Kunden in Kontakt zu treten, Erwartungen zu übertreffen und sie zu Promotoren zu machen, die Ihr Geschäft ausbauen.
- Operations Hub — Betriebssoftware, die Ihre Apps synchronisiert, Kundendaten bereinigt und erstellt und Prozesse automatisiert, sodass all Ihre Systeme und Teams besser zusammenarbeiten.

Themen

- AWS Glue Unterstützung für HubSpot
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration HubSpot
- HubSpot Verbindungen konfigurieren
- Aus HubSpot Entitäten lesen
- HubSpot Verbindungsoptionen
- Einschränkungen und Hinweise für HubSpot Connector

AWS Glue Unterstützung für HubSpot

AWS Glue unterstützt HubSpot wie folgt:

Als Quelle unterstützt?

Ja — Sync und Async. Sie können AWS Glue ETL-Jobs verwenden, um Daten von HubSpot abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte HubSpot API-Versionen

Die folgenden HubSpot API-Versionen werden unterstützt:

- v1
- v2
- v3
- v4

Informationen zur versionsspezifischen Unterstützung von Entitäten finden Sie unter Unterstützte Entitäten für Sync-Quelle undUnterstützte Entitäten für Async-Quelle.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration HubSpot

Bevor Sie Daten von AWS Glue übertragen können HubSpot, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein HubSpot Konto. Weitere Informationen finden Sie unter Ein HubSpot Konto erstellen.
- Ihr HubSpot Konto ist für den API-Zugriff aktiviert.
- Ihr HubSpot Entwicklerkonto sollte über eine App verfügen, die die Kundenanmeldedaten bereitstellt, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Aufrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Eine HubSpot Entwickler-App erstellen.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem HubSpot Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun HubSpot.

Ein HubSpot Konto erstellen

Um ein HubSpot Konto zu erstellen:

1. Gehen Sie zur HubSpot SignUp CRM-URL.
2. Geben Sie Ihre E-Mail-Adresse ein und wählen Sie E-Mail verifizieren (Sie können sich auch mit einem Google-, Microsoft- oder Apple-Konto anmelden).
3. Suchen Sie in Ihrem Posteingang nach dem Bestätigungscode von HubSpot.
4. Geben Sie den sechsstelligen Bestätigungscode ein und klicken Sie auf Weiter.
5. Geben Sie ein Passwort ein und klicken Sie auf Weiter.

6. Geben Sie Ihren Vor- und Nachnamen ein und klicken Sie auf Weiter oder registrieren Sie sich über den Link Mit Google registrieren.
7. Geben Sie Ihre Branche ein und klicken Sie auf Weiter.
8. Geben Sie Ihre berufliche Rolle ein und klicken Sie auf Weiter.
9. Geben Sie Ihren Firmennamen ein und klicken Sie auf Weiter.
10. Wählen Sie die Größe Ihres Unternehmens (Anzahl der in Ihrem Unternehmen beschäftigten Mitarbeiter) und klicken Sie auf Weiter.
11. Geben Sie die Website Ihres Unternehmens ein und klicken Sie auf Weiter.
12. Wählen Sie aus, wo Ihre Daten gehostet werden sollen (Vereinigte Staaten oder Europa), und klicken Sie auf Konto erstellen.
13. Wählen Sie den Zweck der Kontoerstellung aus und klicken Sie auf Weiter.
14. Wählen Sie Google-Konto Connect oder fügen Sie selbst Kontakte hinzu, um Ihre Kontakte mit Ihrem HubSpot Konto zu verknüpfen.
15. Melden Sie sich in Ihrem Google-Konto an, wenn Sie die Option Google-Konto Connect ausgewählt haben, um Ihre Kontakte zu verknüpfen und Ihr HubSpot Konto zu verwenden.

Eine HubSpot Entwickler-App erstellen

App-Entwicklerkonten sind für die Erstellung und Verwaltung von Apps, Integrationen und Entwickler-Testkonten vorgesehen. Sie sind auch der Ort, an dem Sie App Marketplace-Angebote erstellen und verwalten können. App-Entwicklerkonten und die zugehörigen Testkonten sind jedoch nicht mit einem HubSpot Standardkonto verbunden. Sie können keine Daten oder Assets mit oder von einem anderen HubSpot Konto synchronisieren. Um die Client ID und den Client Secret zu erhalten, erstellen Sie ein Entwicklerkonto.

1. Gehe zu https://developers.hubspot.com/
2. Wählen Sie Entwicklerkonto erstellen und scrollen Sie nach unten.
3. Sie werden gefragt, ob Sie ein App-Entwicklerkonto, ein privates App-Konto oder ein CMS-Entwickler-Sandbox-Konto erstellen möchten. Wählen Sie Konto für App-Entwickler erstellen.
4. Da Sie bereits ein Konto bei erstellt haben HubSpot, können Sie Weiter mit diesem Benutzer wählen.
5. Klicken Sie auf Registrierung starten.
6. Geben Sie Ihre Job Rolle ein und klicken Sie auf Weiter.

7. Geben Sie Ihrem Entwicklerkonto einen Namen und klicken Sie auf Weiter und dann auf Überspringen.
8. Wählen Sie Create app (App erstellen).
9. Sobald Ihre App erstellt ist, wählen Sie Auth.
10. Notieren Sie sich unter Auth die Client-ID und das Client-Geheimnis.
11. Fügen Sie Ihre regionsspezifische Weiterleitungs-URL als https: *//<aws-region>* .console.aws.amazon hinzu. com/gluestudio/oauth. For example, add https://us-east-1.console.aws.amazon.com/gluestudio/oauthfür die Region US-East-1.
12. Scrollen Sie nach unten und suchen Sie nach Bereichen. Es gibt zwei Arten von Bereichen, die Sie unter den Überschriften „CRM“ und „Standard“ auswählen müssen.
13. Fügen Sie die folgenden Bereiche hinzu:

```
content
automation
oauth
crm.objects.owners.read
forms
tickets
crm.objects.contacts.write
e-commerce
crm.schemas.custom.read
crm.objects.custom.read
sales-email-read
crm.objects.custom.write
crm.objects.companies.write
crm.lists.write
crm.objects.companies.read
crm.lists.read
crm.objects.deals.read
crm.objects.deals.write
crm.objects.contacts.read
```

14. Klicken Sie auf Speichern und Ihr Entwicklerkonto ist jetzt einsatzbereit.
15. Scrollen Sie nach oben, um die Client-ID zu finden.
16. Klicken Sie auf derselben Seite auf Anzeigen, um den geheimen Client-Schlüssel abzurufen.

Erstellen eines HubSpot Entwickler-Testkontos

In App-Entwicklerkonten können Sie Entwickler-Testkonten erstellen, um Apps und Integrationen zu testen, ohne dass sich dies auf reale HubSpot Daten auswirkt. Testkonten für Entwickler entsprechen nicht den Produktionskonten, sondern haben Zugriff auf eine 90-Tage-Testversion der Enterprise-Versionen von Marketing, Vertrieb, Service, CMS und Operations Hub, sodass Sie die meisten HubSpot Tools testen können und. APIs

1. Klicken Sie auf Home.
2. Klicken Sie auf Testkonto erstellen.
3. Klicken Sie auf App-Testkonto erstellen.
4. Ein neues Fenster wird angezeigt. Geben Sie den Namen des App-Testkontos ein und klicken Sie auf Erstellen.

   Ihr App-Testkonto ist jetzt erstellt.

> **Note**
>
> Das Entwicklerkonto bezieht sich auf Entwicklungsaktivitäten wie die API-Integration, und das App-Testkonto wird verwendet, um die Daten zu sehen, die vom Entwicklerkonto erstellt oder abgerufen werden.

HubSpot Verbindungen konfigurieren

HubSpot unterstützt den Grant-Typ AUTHORIZATION_CODE für. OAuth2

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters umgeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, muss die OAuth zugehörigen Informationen wie Client-ID und Client Secret für seine HubSpot Client-Anwendung angeben. Die AWS Glue Konsole leitet den Benutzer HubSpot dorthin weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine HubSpot Instanz gewähren muss.
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in zu erstellen HubSpot und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin

weitergeleitet, HubSpot um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
- Öffentliche HubSpot Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter Öffentliche Apps.

So konfigurieren Sie eine HubSpot Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für die Verbindung in erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps HubSpot.
   b. Stellen Sie die HubSpot Umgebung bereit.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus HubSpot Entitäten lesen

Voraussetzung

Ein HubSpot Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Kontakt oder Aufgabe. Die folgende Tabelle zeigt die unterstützten Entitäten für Sync-Quelle.

Unterstützte Entitäten für Sync-Quelle

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt die Partitionierung
Kampagnen	v1	Nein	Ja	Nein	Ja	Nein
Unternehmen	v3	Ja	Ja	Ja	Ja	Ja
Kontakte	v3	Ja	Ja	Ja	Ja	Ja
Kontaktlisten	v1	Nein	Ja	Nein	Ja	Nein
Angebote	v3	Ja	Ja	Ja	Ja	Ja
CRM-Pipeline (Angebotspipelines)	v1	Nein	Nein	Nein	Ja	Nein

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt die Partitionierung
E-Mail-Ereignisse	v1	Nein	Ja	Nein	Ja	Nein
Calls	v3	Ja	Ja	Ja	Ja	Ja
Hinweise	v3	Ja	Ja	Ja	Ja	Ja
E-Mails	v3	Ja	Ja	Ja	Ja	Ja
Treffen	v3	Ja	Ja	Ja	Ja	Ja
Aufgaben	v3	Ja	Ja	Ja	Ja	Ja
Postsendungen	v3	Ja	Ja	Ja	Ja	Ja
Benutzerdefinierte Objekte	v3	Ja	Ja	Ja	Ja	Ja
Formen	v2	Nein	Nein	Nein	Ja	Nein
Eigentümer	v3	Nein	Ja	Nein	Ja	Nein
Produkte	v3	Ja	Ja	Ja	Ja	Ja
Eintrittskarten	v3	Ja	Ja	Ja	Ja	Ja
Workflows	v3	Nein	Nein	Nein	Ja	Nein
Zuordnungen	v4	Ja	Nein	Nein	Ja	Nein

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt die Partitionierung
Labels für Verbände	v4	Nein	Nein	Nein	Ja	Nein

Beispiel:

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "contact",
        "API_VERSION": "v3"
    }
```

Unterstützte Entitäten für Async-Quelle

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt die Partitionierung
Unternehmen	v3	Ja	Nein	Ja	Ja	Nein
Kontakte	v3	Ja	Nein	Ja	Ja	Nein
Angebote	v3	Ja	Nein	Ja	Ja	Nein
Calls	v3	Ja	Nein	Ja	Ja	Nein
Hinweise	v3	Ja	Nein	Ja	Ja	Nein
E-Mails	v3	Ja	Nein	Ja	Ja	Nein
Treffen	v3	Ja	Nein	Ja	Ja	Nein

Entität	API-Version	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt die Partitionierung
Aufgaben	v3	Ja	Nein	Ja	Ja	Nein
Postsendungen	v3	Ja	Nein	Ja	Ja	Nein
Benutzerdefinierte Objekte	v3	Ja	Nein	Ja	Ja	Nein
Produkte	v3	Ja	Nein	Ja	Ja	Nein
Eintrittskarten	v3	Ja	Nein	Ja	Ja	Nein

Beispiel:

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "contact",
        "API_VERSION": "v3",
        "TRANSFER_MODE": "ASYNC"
    }
```

HubSpot Entitäts- und Felddetails:

HubSpot API v4:

Entität	API-Version	Feld	Datentyp	Unterstützte Operatoren
Bezeichnung des Vereins	v4	Kategorie	String	N/A

Entität	API-Version	Feld	Datentyp	Unterstützte Operatoren
		typeId	Ganzzahl	N/A
		Bezeichnung	String	N/A
Zuordnungen		from	Struct	N/A
		id	String	"="
		to	Auflisten	N/A

> **Note**
>
> Um Assoziationen zwischen zwei Objekten für das `Associations` Objekt abzurufen, müssen Sie beim Erstellen eines Jobs die Absender-ID (die ID des ersten Objekts) über einen obligatorischen Filter angeben. AWS Glue Wenn Sie in diesem Fall Assoziationen für mehrere abrufen möchten, müssen Sie IDs in der Klausel mehrere IDs angeben. `where` Zum Beispiel: Zum Abrufen `Associations` der Kontakte IDs '1' und '151' müssen Sie einen Filter als angeben. `where id=1 AND id=151`

HubSpot API v3:

Entität	Feld	Datentyp	Unterstützte Operatoren
Eigentümer	firstName	String	N/A
	lastName	String	N/A
	createdAt	DateTime	N/A
	archived	Boolesch	N/A
	Teams	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	id	String	N/A
	userId	Ganzzahl	N/A
	email	String	N/A
	updatedAt	DateTime	N/A
Workflow	Name	String	N/A
	id	Ganzzahl	N/A
	Typ	String	N/A
	aktiviert	Boolesch	N/A
	Eingefügt bei	Long	N/A
	updatedAt	Long	N/A
	contactListIds	Struct	N/A
	personaTagIds	Auflisten	N/A

HubSpot Stellt für die folgenden Entitäten Endpunkte zum dynamischen Abrufen von Metadaten bereit, sodass die Operatorunterstützung für jede Entität auf Datentypebene erfasst wird.

> **Note**
>
> `DML_STATUS`ist ein virtuelles Feld, das jedem Datensatz zur Laufzeit hinzugefügt wird, um seinen Status (ERSTELLT/AKTUALISIERT) im Synchronisierungsmodus zu bestimmen. Der `CONTAINS/LIKE` Operator wird im asynchronen Modus nicht unterstützt.

Entität	Datentyp	Unterstützte Operatoren
Kontakt	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Unternehmen	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Deal	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A

Entität	Datentyp	Unterstützte Operatoren
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Fahrkarte	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Produkt	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A

Entität	Datentyp	Unterstützte Operatoren
Benutzerdefiniertes Objekt	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Rufen Sie	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Email	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A

Entität	Datentyp	Unterstützte Operatoren
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Meeting	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Hinweis	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A

Entität	Datentyp	Unterstützte Operatoren
Aufgabe	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A
Postalische Post	Ganzzahl	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	„=,! =, WIE“
	Datum	N/A
	DateTime	„zwischen“
	Boolesch	"="
	Auflisten	N/A
	Struct	N/A

HubSpot API v2:

Entität	Feld	Datentyp	Unterstützte Operatoren
Formular	Portal-ID	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Leitfaden	String	N/A
	Name	String	N/A
	Methode	String	N/A
	CSS-Klasse	String	N/A
	umleiten	String	N/A
	Text abschicken	String	N/A
	Empfänger benachrichtigen	String	N/A
	createdAt	Long	N/A
	updatedAt	Long	N/A
	ignoreCurrentValues	Boolesch	N/A
	löschbar	Boolesch	N/A
	Inline-Nachricht	Boolesch	N/A
	Captcha aktiviert	Boolesch	N/A
	klonbar	Boolesch	N/A
	formFieldGroups	Auflisten	N/A
	editable	Boolesch	N/A
	Gelöscht am	Ganzzahl	N/A
	Name des Themas	String	N/A
	ID des Elternteils	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Stil	String	N/A
	isPublished	Boolesch	N/A
	Veröffentlichen bei	Ganzzahl	N/A
	Veröffentlichung rückgängig machen	Ganzzahl	N/A
	Veröffentlicht unter	Ganzzahl	N/A
	kickbackEmailWorkflowId	String	N/A
	kickbackEmailsJson	Ganzzahl	N/A
	Benutzerdefinierte UID	String	N/A
	createMarketableContact	Boolesch	N/A
	Version bearbeiten	Ganzzahl	N/A
	thankYouMessageJson	String	N/A
	Farbe des Themas	String	N/A
	alwaysCreateNewFirma	Boolesch	N/A
	internalUpdatedAt	Long	N/A
	businessUnitId	Ganzzahl	N/A
	Tragbarer Schlüssel	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	paymentSessionTemplateIDs	Auflisten	N/A
	selectedExternalOptions	Auflisten	N/A

HubSpot API v1:

Entität	Feld	Datentyp	Unterstützte Operatoren
Kampagne	id	Ganzzahl	N/A
	appId	Ganzzahl	N/A
	Name der App	String	N/A
	lastUpdatedTime	Long	N/A
Kontaktliste	dynamisch	Boolesch	N/A
	Name	String	N/A
	Portal-ID	Ganzzahl	N/A
	createdAt	Long	N/A
	listId	Ganzzahl	N/A
	updatedAt	Long	N/A
	ListType	String	N/A
	-Filter	Auflisten	N/A
	Autoren-ID	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Metadaten	Struct	N/A
	archived	Boolesch	N/A
	ilsFilterBranch	String	N/A
	IDs filtern	Auflisten	N/A
	LimitExempt	Boolesch	N/A
	intern	Boolesch	N/A
	readOnly	Boolesch	N/A
	ID der Eltern	Ganzzahl	N/A
E-Mail_Ereignis	id	String	N/A
	Typ	String	N/A
	Empfänger	String	N/A
	Portal-ID	Ganzzahl	N/A
	appId	Ganzzahl	N/A
	Name der App	String	N/A
	emailCampaignId	Long	N/A
	versuchen	Ganzzahl	N/A
	created	Long	N/A
	Gesendet von	Struct	N/A
	SMTP-ID	String	N/A
	response	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	subject	String	N/A
	cc	Auflisten	N/A
	bcc	Auflisten	N/A
	Antworte auf	Auflisten	N/A
	from	String	N/A
	Grund fallen lassen	String	N/A
	Nachricht löschen	String	N/A
	Browser	Struct	N/A
	userAgent	String	N/A
	duration	Long	N/A
	location	Struct	N/A
	Gefiltertes Ereignis	Boolesch	N/A
	Typ des Geräts	String	N/A
	Unterdrückter Grund	String	N/A
	Unterdrückte Nachricht	String	N/A
CRM_Pipeline	Pipeline-ID	String	N/A
	createdAt	Long	N/A
	updatedAt	Long	N/A
	objectType	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Bezeichnung	String	N/A
	Reihenfolge anzeigen	Ganzzahl	N/A
	aktiv	Boolesch	N/A
	Stufen	Auflisten	N/A
	objectTypeId	String	N/A
	default	Boolesch	N/A

Abfragen partitionieren

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiele für gültige Werte:

  ```
  “2024-01-01T10:00:00.115Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

In der folgenden Tabelle werden die Informationen zur Unterstützung von Feldern zur Partitionierung von Entitäten beschrieben:

Entity-Name	Partitionierung von Feldern	Datentyp
contact	hs_object_id	Long
	Erstellungsdatum, Datum der letzten Änderung	DateTime
company	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Deal	hs_object_id	Long
	Erstellungsdatum, hs_createdate, hs_lastmodifieddate	DateTime
Fahrkarte	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Produkt	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Benutzerdefiniertes_Objekt	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Aufruf	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
email	hs_object_id	Long

Entity-Name	Partitionierung von Feldern	Datentyp
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Treffen	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
note	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
Aufgabe	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime
postal_mail	hs_object_id	Long
	Erstellungsdatum, hs_lastmodifieddate	DateTime

Beispiel:

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "v3",
        "PARTITION_FIELD": "hs_object_id"
        "LOWER_BOUND": "50"
        "UPPER_BOUND": "16726619290"
        "NUM_PARTITIONS": "10"
    }
```

HubSpot Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für HubSpot:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in HubSpot.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. HubSpot Rest-API-Version, die Sie verwenden möchten. Zum Beispiel: v1, v2, v3, v4.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `TRANSFER_MODE`(String) — Wird verwendet, um anzugeben, ob die Abfrage im asynchronen Modus ausgeführt werden soll.

Einschränkungen und Hinweise für HubSpot Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den HubSpot Konnektor:

- Die Suchendpunkte sind auf insgesamt 10.000 Ergebnisse für eine bestimmte Abfrage begrenzt. Jede Partition mit mehr als 10.000 Datensätzen führt zu einem 400-Fehler.
- Weitere wichtige Einschränkungen für den Connector werden unter Einschränkungen beschrieben.
- Maximal drei Filteranweisungen werden von akzeptiert HubSpot.
- HubSpot Unterstützt derzeit Verknüpfungen zwischen HubSpot Standardobjekten (z. B. Kontakt, Firma, Geschäft oder Ticket) und benutzerdefinierten Objekten.

  - Für ein kostenloses Konto: Sie können nur bis zu 10 Zuordnungstypen zwischen jeder Objektpaarung (z. B. Kontakte und Unternehmen) erstellen.
  - Für ein Super-Admin-Konto: Sie können nur bis zu 50 Zuordnungstypen zwischen jeder Objektpaarung erstellen.
  - Weitere Informationen finden Sie unter Assoziationen v4 und Assoziationsbezeichnungen erstellen und verwenden.
- Die Objekte „Quote“ und „Communications“ sind für Associations nicht vorhanden, da sie derzeit im Connector nicht unterstützt werden.
- Für Async sortiert SaaS die Werte nur in aufsteigender Reihenfolge.
- Für die `Ticket` Entität gibt SaaS das `hs_object_id` Feld nicht im asynchronen Modus zurück.

Verbindung zu Instagram-Anzeigen herstellen

Instagram ist eine beliebte App zum Teilen von Fotos, mit der Sie mit Marken, Prominenten, Vordenkern, Freunden, Familienmitgliedern und mehr in Kontakt treten können. Es ist ein Dienst zum Teilen von Fotos und für soziale Netzwerke. Benutzer können Fotos oder kurze Videos aufnehmen und diese mit ihren Followern teilen. Instagram-Anzeigen sind Beiträge, für die Unternehmen bezahlen können, um sie Instagram-Nutzern zur Verfügung zu stellen.

Themen

- AWS Glue Unterstützung für Instagram-Anzeigen
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Instagram-Anzeigen
- Konfiguration von Instagram Ads-Verbindungen
- Lesen von Instagram Ads-Entitäten
- Verbindungsoptionen für Instagram-Anzeigen
- Einschränkungen und Hinweise für den Instagram Ads Connector

AWS Glue Unterstützung für Instagram-Anzeigen

AWS Glue unterstützt Instagram-Anzeigen wie folgt:

Als Quelle unterstützt?

Ja. Du kannst AWS Glue ETL-Jobs verwenden, um Daten aus Instagram-Anzeigen abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen von Instagram Ads

Die folgenden Instagram Ads API-Versionen werden unterstützt:

- v17.0
- v18.0
- v19.0
- v20.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Instagram-Anzeigen

Bevor du Daten von Instagram Ads übertragen kannst, musst du die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Auf Instagram-Standardkonten wird indirekt über Facebook zugegriffen.
- Zur Generierung des Zugriffstokens ist eine Benutzerauthentifizierung erforderlich.
- Der Instagram Ads SDK-Connector wird den User Access OAuth Token-Flow implementieren.
- Wir verwenden OAuth2 2.0, um unsere API-Anfragen für Instagram-Anzeigen zu authentifizieren. Diese webbasierte Authentifizierung fällt unter die Multi-Factor Authentication (MFA) -Architektur, die eine Obergruppe von 2FA darstellt.
- Der Benutzer muss Berechtigungen für den Zugriff auf die Endpunkte gewähren. Für den Zugriff auf die Benutzerdaten erfolgt die Autorisierung der Endgeräte über Berechtigungen und Funktionen.

Anmeldeinformationen für OAuth Version 2.0 abrufen

Informationen zum Abrufen von API-Anmeldeinformationen, mit denen Sie authentifizierte Aufrufe an Ihre Instance tätigen können, finden Sie unter Graph-API.

Konfiguration von Instagram Ads-Verbindungen

Instagram Ads unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2

- Dieser Zuschusstyp wird als dreibeiniges eingestuft, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Instagram Ads zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben. In diesem Szenario werden sie weiterhin zu Instagram Ads weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Zugriffstoken. Ein ablaufendes Systembenutzertoken ist ab einem generierten oder aktualisierten Datum 60 Tage lang gültig. Um Kontinuität zu gewährleisten, sollte der Entwickler das Zugriffstoken innerhalb von 60 Tagen aktualisieren. Andernfalls verfällt das Zugriffstoken und der Entwickler muss ein neues erwerben, um den API-Zugriff wiederherzustellen. Siehe Zugriffstoken aktualisieren.

So konfigurierst du eine Instagram Ads-Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Instagram Ads aus.
   b. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

c. Geben Sie die Client-ID der vom Benutzer verwalteten Client-Anwendung ein.

d. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen. Das ausgewählte Geheimnis muss über einen Schlüssel `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` verfügen, dessen Wert dem geheimen Clientschlüssel der verbundenen App entspricht.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen von Instagram Ads-Entitäten

Voraussetzung

Ein Instagram Ads-Objekt, aus dem du lesen möchtest. Du benötigst den Objektnamen. Die folgenden Tabellen zeigen die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kampagne	Ja	Ja	Nein	Ja	Ja
Anzeigenset	Ja	Ja	Nein	Ja	Ja
Werbeanzeigen	Ja	Ja	Nein	Ja	Ja
Und kreativ	Nein	Ja	Nein	Ja	Nein
Insights — Konto	Nein	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Bild hinzufügen	Ja	Ja	Nein	Ja	Nein
Einblicke — Anzeige	Ja	Ja	Nein	Ja	Ja
Einblicke - AdSet	Ja	Ja	Nein	Ja	Ja
Einblicke — Kampagne	Ja	Ja	Nein	Ja	Ja

Beispiel:

```
instagramAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="instagramads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0"
    }
```

Entitäts- und Felddetails für Instagram Ads

Weitere Informationen zu den Entitäten und Felddetails finden Sie unter:

- Kampagne
- Anzeigen-Set
- Anzeige
- Und kreativ
- Einblick in das Werbekonto
- Bild hinzufügen
- Einblicke in Werbeanzeigen

- AdSets Einblicke
- Einblicke in Kampagnen

Weitere Informationen finden Sie unter Marketing-API.

> **Note**
>
> Die Datentypen Struct und List werden in der Antwort der Konnektoren in den Datentyp String konvertiert.

Abfragen partitionieren

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiel für einen gültigen Wert:

  ```
  "2022-01-01T00:00:00.000Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.

  Beispiel für einen gültigen Wert:

  ```
  "2024-01-02T00:00:00.000Z"
  ```

- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
instagramADs_read = glueContext.create_dynamic_frame.from_options(
    connection_type="instagramads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0",
        "PARTITION_FIELD": "created_time"
        "LOWER_BOUND": "2022-01-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-01-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Instagram-Anzeigen

Im Folgenden sind die Verbindungsoptionen für Instagram-Anzeigen aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Instagram-Anzeigen.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die API-Version von Instagram Ads Graph, die Sie verwenden möchten. Zum Beispiel: v21.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen und Hinweise für den Instagram Ads Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Instagram Ads Connector:

- Die Anzahl der Anrufe einer App ist die Anzahl der Anrufe, die ein Benutzer innerhalb eines fortlaufenden einstündigen Zeitfensters tätigen kann, 200 multipliziert mit der Anzahl der Benutzer. Einzelheiten zu den Ratenlimits finden Sie unter Ratenlimits und Ratenlimits für geschäftliche Anwendungsfälle.

Verbindung zur Gegensprechanlage herstellen in AWS Glue Studio

Intercom ist das Engagement OS, ein offener Kanal zwischen Ihrem Unternehmen und Ihren Kunden — in Bezug auf das Produkt, im Moment und zu deren Bedingungen. Er ermöglicht einen kontinuierlichen Dialog, der es Ihnen ermöglicht, das Beste aus jeder Interaktion während der gesamten Kundenreise herauszuholen.

Themen

- AWS Glue Unterstützung für Intercom
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Intercom konfigurieren
- Intercom-Verbindungen konfigurieren
- Lesen von Intercom-Entitäten
- Intercom-Verbindungsoptionen
- Einschränkungen
- Neues Intercom-Konto erstellen und die Client-App konfigurieren

AWS Glue Unterstützung für Intercom

AWS Glue unterstützt Intercom wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Intercom abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Intercom-API-Versionen

v2.5. Informationen zur versionsspezifischen Unterstützung von Entitäten finden Sie unterLesen von Intercom-Entitäten.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM und Amazon EC2. Amazon CloudWatch Logs Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie

die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Intercom konfigurieren

Bevor Sie die Übertragung von Intercom verwenden AWS Glue können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein Intercom-Konto. Weitere Informationen finden Sie unter Neues Intercom-Konto erstellen und die Client-App konfigurieren.
- Ihr Intercom-Konto ist für den API-Zugriff aktiviert.
- Unter dem Intercom-Entwicklerkonto sollten Sie über eine App verfügen, die die Kundenanmeldedaten bereitstellt, mit AWS Glue denen Sie sicher auf Ihre Daten zugreifen können, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Intercom — Neues Konto und Schritte zur Erstellung einer Kunden-App.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Intercom-Konto herstellen.

Intercom-Verbindungen konfigurieren

Intercom unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für 2. OAuth

Dieser Zuweisungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Nutzer zu Google Ads weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Intercom-Instanz gewähren muss.

Benutzer sollten beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihr eigenes Client-Geheimnis angeben. In diesem Szenario werden sie weiterhin zu Intercom weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.

Weitere Informationen zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss findest du unter Ads-API.

So konfigurieren Sie eine Intercom-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps — Secret sollte das Zugriffstoken für die verbundene App, das Aktualisierungstoken, client_id und client_secret enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Intercom.
   b. Stellen Sie die Intercom-Umgebung bereit.
   c. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Lesen von Intercom-Entitäten

Voraussetzungen

- Ein Intercom-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	API_VVersion	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Administratoren	v2.5	Nein	Nein	Nein	Ja	Nein
Unternehmen	v2.5	Nein	Ja	Nein	Ja	Nein
Konversationen	v2.5	Ja	Ja	Ja	Ja	Ja
Datenattribute	v2.5	Nein	Nein	Nein	Ja	Nein
Kontakte	v2.5	Ja	Ja	Ja	Ja	Ja
Segmente	v2.5	Nein	Nein	Nein	Ja	Nein
Tags	v2.5	Nein	Nein	Nein	Ja	Nein
Teams	v2.5	Nein	Nein	Nein	Ja	Nein

Beispiel

```
Intercom_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Intercom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "V2.5"
    }
)
```

Angaben zur Intercom-Entität und zum Feld

Entität	Feld	Datentyp	Unterstützte Operatoren
Administratoren	Typ	String	N/A
Administratoren	id	String	N/A
Administratoren	Avatar	Struct	N/A
Administratoren	Name	String	N/A
Administratoren	email	String	N/A
Administratoren	away_mode_aktiviert	Boolesch	N/A
Administratoren	away_mode_reassign	Boolesch	N/A
Administratoren	hat_posteingang_sitz	Boolesch	N/A
Administratoren	Teams-IDs	Auflisten	N/A
Administratoren	Berufsbezeichnung	String	N/A
Unternehmen	Typ	String	N/A
Unternehmen	id	String	N/A
Unternehmen	app_id	String	N/A
Unternehmen	created_at	DateTime	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Unternehmen	remote_created_at	DateTime	N/A
Unternehmen	updated_at	DateTime	N/A
Unternehmen	last_request_at	DateTime	N/A
Unternehmen	Planen	Struct	N/A
Unternehmen	company_id	String	N/A
Unternehmen	Name	String	N/A
Unternehmen	custom_attributes	Struct	N/A
Unternehmen	session_count	Ganzzahl	N/A
Unternehmen	monthly_spend	Ganzzahl	N/A
Unternehmen	user_count	Ganzzahl	N/A
Unternehmen	Industrie	String	N/A
Unternehmen	size	Ganzzahl	N/A
Unternehmen	website	String	N/A
Unternehmen	tags	Struct	N/A
Unternehmen	Segmente	Struct	N/A
Kontakte	id	String	GLEICH_ZU .NICHT_GLEICH_
Kontakte	Typ	String	N/A
Kontakte	Arbeitsbereichs-ID	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontakte	external_id	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	role	String	GLEICH_ZU .NICHT_GLEICH_ZU
Kontakte	email	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	phone	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Name	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Avatar	String	N/A
Kontakte	owner_id	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	soziale_profile	Struct	N/A
Kontakte	ist schwer abgeprallt	Boolesch	EQUAL_TO
Kontakte	E-Mail als Spam markiert	Boolesch	EQUAL_TO
Kontakte	hat sich von E-Mails abgemeldet	Boolesch	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontakte	created_at	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	updated_at	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	angemeldet_am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	zuletzt gesehen am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	zuletzt geantwortet am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	zuletzt kontaktiert am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	Letzte_E-Mail_geöffnet_am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	letzte_E-Mail_angeklickt_am	DateTime	GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Kontakte	sprache_überschreiben	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontakte	Browser	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Browserversion	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Browsersprache	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	os	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	location	Struct	N/A
Kontakte	Standort_Land	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	standort_region	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Standort_Stadt	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	Name der Android-App	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	android_app_version	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontakte	android_gerät	String	N/A
Kontakte	android_os_version	String	N/A
Kontakte	android_sdk-version	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	android_last_seen_at	Datum	N/A
Kontakte	iOS-App-Name	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	ios_app_version	String	N/A
Kontakte	ios_gerät	String	N/A
Kontakte	ios_os_version	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	ios_sdk_version	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Kontakte	ios_last_seen_at	DateTime	N/A
Kontakte	benutzerdefinierte _Attribute	Struct	N/A
Kontakte	tags	Struct	N/A
Kontakte	notes	Struct	N/A
Kontakte	Unternehmen	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Kontakte	abmelden_von_SMS	Boolesch	N/A
Kontakte	sms_consent	Boolesch	N/A
Kontakte	Abonnementtypen, für die Sie sich entschied en haben	Struct	N/A
Kontakte	Referrer	String	N/A
Kontakte	utm_campaign	String	N/A
Kontakte	utm_content	String	N/A
Kontakte	utm_medium	String	N/A
Kontakte	utm_source	String	N/A
Kontakte	utm_term	String	N/A
Konversationen	Typ	String	N/A
Konversationen	id	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	created_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	updated_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	Quelle	Struct	N/A
Konversationen	source_id	String	GLEICH_ZU, NICHT_GLEICH_ZU
Konversationen	source_type	String	EQUAL_TO, NOT_EQUAL_TO,
Konversationen	source_delivered_as	String	GLEICH_ZU, NICHT_GLEICH_ZU,
Konversationen	source_subject	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	source_body	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	source_author_id	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	source_author_type	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	source_author_name	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	source_author_email	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	source_url	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	Kontakte	Struct	N/A
Konversationen	Teammitglieder	Struct	N/A
Konversationen	Titel	String	N/A
Konversationen	admin_assignee_id	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	team_assignee_id	Ganzzahl	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	custom_attributes	Struct	N/A
Konversationen	geöffnet	Boolesch	EQUAL_TO
Konversationen	state	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	read	Boolesch	EQUAL_TO
Konversationen	warten_seit	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	snoozed_until	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	tags	Struct	N/A
Konversationen	first_contact_reply	Struct	N/A
Konversationen	priority	String	EQUAL_TO, NOT_EQUAL_TO
Konversationen	topics	Struct	N/A
Konversationen	sla_angewendet	Struct	N/A
Konversationen	Konversationsbewertung	Struct	N/A
Konversationen	conversation_rating_requested_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	conversation_rating_replied_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	conversation_rating_score	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	Bemerkung zur Konversation	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	conversation_rating_contact_id	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	conversation_rating_admin_id	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	statistics	Struct	N/A
Konversationen	statistics_time_to_assignment	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_time_to_admin_reply	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_time_to_first_close	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_time_to_last_close	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	statistics_median_time_to_reply	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_first_contact_reply_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_first_assignment_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_first_admin_reply_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_first_close_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_last_assignment_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_last_assignment_admin_reply_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen	statistics_last_contact_reply_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_last_admin_reply_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_last_close_at	DateTime	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_last_closed_by_id	String	ENTHÄLT, EQUAL_TO, NOT_EQUAL_TO
Konversationen	statistics_count_wird erneut geöffnet	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_count_assignments	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	statistics_count_conversation_parts	Ganzzahl	GLEICH_ALS, NICHT_GLEICH_ALS, GRÖSSER_ALS, KLEINER_ALS
Konversationen	Konversationsteile	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Datenattribute	id	Ganzzahl	N/A
Datenattribute	Typ	String	N/A
Datenattribute	model	String	N/A
Datenattribute	Name	String	N/A
Datenattribute	vollständiger_Name	String	N/A
Datenattribute	Bezeichnung	String	N/A
Datenattribute	description	String	N/A
Datenattribute	data_type	String	N/A
Datenattribute	options	Auflisten	N/A
Datenattribute	api_writable	Boolesch	N/A
Datenattribute	ui_writable	Boolesch	N/A
Datenattribute	benutzerdefiniert	Boolesch	N/A
Datenattribute	archived	Boolesch	N/A
Datenattribute	created_at	Boolesch	N/A
Datenattribute	updated_at	DateTime	N/A
Datenattribute	admin_id	String	N/A
Segmente	Typ	String	N/A
Segmente	id	String	N/A
Segmente	Name	String	N/A
Segmente	created_at	DateTime	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Segmente	updated_at	DateTime	N/A
Segmente	Personentyp	String	N/A
Segmente	count	Ganzzahl	N/A
Tags	Typ	String	N/A
Tags	id	String	N/A
Tags	Name	String	N/A
Teams	Typ	String	N/A
Teams	id	String	N/A
Teams	Name	String	N/A
Teams	Admin-IDs	Auflisten	N/A

Abfragen partitionieren

Zusätzliche Spark-OptionenPARTITION_FIELD,LOWER_BOUND,UPPER_BOUND, NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:"2024-02-06".
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt.

Entity Name	Feld partitionieren	Datentyp
Kontakte	created_at, updated_at, last_seen_at	DateTime
Konversationen	id	Ganzzahl
Konversationen	created_at, updated_at	DateTime

Beispiel

```
Intercom_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Intercom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversation",
        "API_VERSION": "V2.5",
        "PARTITION_FIELD": "created_at"
        "LOWER_BOUND": "2022-07-13T07:55:27.065Z"
        "UPPER_BOUND": "2022-08-12T07:55:27.065Z"
        "NUM_PARTITIONS": "2"
    }
)
```

Intercom-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Intercom aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Intercom.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Intercom Rest API-Version, die Sie verwenden möchten. Beispiel: v2.5.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.

- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.
- `INSTANCE_URL`(String) — URL der Instanz, in der der Benutzer die Operationen ausführen möchte. Zum Beispiel: https://api.intercom.io.

Einschränkungen

Die folgenden Einschränkungen gelten für den Intercom-Anschluss:

- Wenn Sie die Entität „Unternehmen“ verwenden, können maximal 10.000 Unternehmen zurückgegeben werden. Weitere Informationen finden Sie unter API „Alle Unternehmen auflisten“.
- Bei der Anwendung von „Reihenfolge nach“ ist der Filter sowohl für die Entitäten „Kontakt“ als auch für „Konversation“ obligatorisch.
- MCA wird vom SaaS-Anbieter unterstützt. Aufgrund der in der Dokumentation genannten API-Ratenbegrenzungen werden wir MCA jedoch nicht hosten, AWS Glue da sich dies auf andere Workloads auswirken und aufgrund von Ressourcenengpässen möglicherweise zu Leistungsproblemen führen kann.

Neues Intercom-Konto erstellen und die Client-App konfigurieren

Ein Intercom-Konto erstellen

1. Wählen Sie die Intercom-URL und dann rechts oben auf der Seite die Option Meine kostenlose Testversion starten aus.
2. Wählen Sie in der rechten oberen Ecke der Seite die Schaltfläche Kostenlos testen.
3. Wählen Sie den Geschäftstyp, den Sie benötigen.
4. Geben Sie alle erforderlichen Informationen auf der Seite ein.

5. Nachdem Sie alle Informationen eingegeben haben, wählen Sie Registrieren.

Eine Intercom-Entwickler-App erstellen

Um die Client ID und den Client Secret zu erhalten, erstellen Sie ein Entwicklerkonto.

1. Navigieren Sie zu https://app.intercom.com/.
2. Geben Sie die E-Mail-ID und das Passwort ein/Melden Sie sich mit Google an und melden Sie sich an.
3. Wählen Sie in der linken unteren Ecke das Benutzerprofil und dann Einstellungen.
4. Wählen Sie Apps & Integration.
5. Wählen Sie unter Apps und Integration den Tab Developer Hub aus.
6. Wählen Sie Neue App und erstellen Sie die App hier.
7. Geben Sie den Namen der App ein und wählen Sie App erstellen.
8. Navigieren Sie in der App zum Abschnitt Authentifizierung.
9. Wählen Sie „Weiterleitung bearbeiten“ und „Weiterleitung hinzufügen URIs“.
   Fügen Sie Ihre regionsspezifische Weiterleitungs-URL als hinzu. `https://<aws-region>.console.aws.amazon.com/gluestudio/oauth` Fügen Sie beispielsweise hinzu. `https://us-east-1.console.aws.amazon.com/gluestudio/oauth for the us-east-1 region`
10. Rufen Sie die generierte Client-ID und das Client-Geheimnis im Abschnitt mit den Basisinformationen ab.

Verbindung zu Jira Cloud herstellen

Jira Cloud ist eine von Atlassian entwickelte Plattform. Die Plattform umfasst Produkte zur Problemverfolgung, mit denen Teams ihre Agile-Projekte planen und verfolgen können. Als Jira Cloud-Nutzer enthält dein Konto Daten zu deinen Projekten, wie z. B. Probleme, Workflows und Ereignisse. Sie können AWS Glue es verwenden, um Ihre Jira Cloud-Daten an bestimmte AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Jira Cloud
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

- Jira Cloud konfigurieren
- Konfiguration von Jira Cloud-Verbindungen
- Aus Jira Cloud-Entitäten lesen
- Verbindungsoptionen für Jira Cloud
- Einschränkungen und Hinweise für Jira Cloud Connector

AWS Glue Unterstützung für Jira Cloud

AWS Glue unterstützt Jira Cloud wie folgt:

Als Quelle unterstützt?

Ja. Du kannst AWS Glue ETL-Jobs verwenden, um Daten aus Jira Cloud abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte Jira Cloud API-Versionen

Die folgenden Jira Cloud API-Versionen werden unterstützt:

- v2

Informationen zur versionsspezifischen Unterstützung von Entitäten findest du unter Unterstützte Entitäten für Source.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Jira Cloud konfigurieren

Bevor du Daten von Jira Cloud AWS Glue an unterstützte Ziele übertragen kannst, musst du die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Du hast ein Atlassian-Konto, mit dem du das Jira-Softwareprodukt in Jira Cloud verwendest. Weitere Informationen finden Sie unter Ein Jira Cloud-Konto erstellen.
- Du musst ein AWS Konto mit dem Servicezugriff auf erstellt haben. AWS Glue

- Diese App stellt die Kundenanmeldedaten bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen findest du unter Enabling OAuth 2.0 (3LO) in der Atlassian Developer-Dokumentation.

Wenn du diese Anforderungen erfüllst, bist du bereit, dich mit deinem Jira AWS Glue Cloud-Konto zu verbinden.

Ein Jira Cloud-Konto erstellen

Um ein Jira Cloud-Konto zu erstellen:

1. Navigiere zur Atlassian-Anmelde-URL.
2. Gib deine geschäftliche E-Mail-Adresse und deinen Namen ein und wähle Zustimmen aus. Sie erhalten eine Bestätigungs-E-Mail.
3. Nachdem Sie Ihre E-Mail-Adresse bestätigt haben, können Sie ein Passwort erstellen und Anmelden wählen.
4. Geben Sie den Namen und das Passwort ein und wählen Sie Anmelden.
5. Sie werden auf eine Seite weitergeleitet, auf der Sie Ihre Site eingeben müssen. Geben Sie einen Site-Namen ein und wählen Sie Zustimmen.

Sobald deine Atlassian Cloud-Site gestartet ist, kannst du deine Jira einrichten, indem du einige Fragen gemäß deinen Einstellungen für den Projekttyp beantwortest.

So meldest du dich bei einem bestehenden Konto an:

1. Navigiere zur Atlassian-Anmelde-URL und gib deine Anmeldedaten ein.
2. Gib E-Mail-Adresse und Passwort ein und klicke auf Anmelden. Sie werden zum Jira-Dashboard weitergeleitet.

Eine App in deiner Jira Cloud erstellen

Um eine App in Jira Cloud zu erstellen und die Client-ID und das Client-Geheimnis von der verwalteten Client-App abzurufen:

1. Navigiere zur Jira Cloud-URL und gib deine Anmeldedaten ein.
2. Wählen Sie Erstellen und wählen Sie die OAuth 2.0-Integrationsoption.

3. Geben Sie den Namen der App ein, überprüfen Sie die Nutzungsbedingungen und wählen Sie Erstellen.
4. Navigiere im linken Menü zum Abschnitt Vertrieb und wähle Bearbeiten.
5. Gehen Sie im Abschnitt Verteilungskontrollen bearbeiten wie folgt vor:
   a. Wählen Sie VERTRIEBSSTATUS als Teilen aus.
   b. Geben Sie den Namen des Anbieters ein.
   c. Geben Sie die URL für Ihre Datenschutzrichtlinie ein. Zum Beispiel https:// docs.aws.amazon.com/glue/ latest/dg/security -iam-awsmanpol.html
   d. Geben Sie die URL für Ihre Nutzungsbedingungen ein (optional).
   e. Geben Sie die URL für Ihren Kundensupport-Kontakt ein (optional).
   f. Wählen Sie in der ERKLÄRUNG ZU PERSÖNLICHEN DATEN die Option Ja/Nein aus und wählen Sie Änderungen speichern aus.
6. Navigiere im linken Menü der jeweiligen App zu Berechtigungen.
7. Wählen Sie für Jira API die Option Hinzufügen aus. Wählen Sie nach dem Hinzufügen die Option Konfiguration.
8. Wähle im Abschnitt Klassische Bereiche > Jira-Plattform REST API die Option Bereiche bearbeiten aus und markiere alle Bereiche. Klicken Sie auf Speichern.
9. Wählen Sie unter „Granular Scopes“ die Option „Bereiche bearbeiten“ und wählen Sie die folgenden Bereiche aus:
10. Scrollen Sie nach unten und suchen Sie nach Bereichen. Es gibt zwei Arten von Bereichen, die Sie unter den Überschriften „CRM“ und „Standard“ auswählen müssen.
11. Fügen Sie die folgenden Bereiche hinzu:

```
read:application-role:jira
read:audit-log:jira
read:avatar:jira
read:field:jira
read:group:jira
read:instance-configuration:jira
read:issue-details:jira
read:issue-event:jira
read:issue-link-type:jira
read:issue-meta:jira
read:issue-security-level:jira
read:issue-security-scheme:jira
read:issue-type-scheme:jira
```

```
read:issue-type-screen-scheme:jira
read:issue-type:jira
read:issue.time-tracking:jira
read:label:jira
read:notification-scheme:jira
read:permission:jira
read:priority:jira
read:project:jira
read:project-category:jira
read:project-role:jira
read:project-type:jira
read:project-version:jira
read:project.component:jir
read:project.property:jira
read:resolution:jira
read:screen:jira
read:status:jira
read:user:jira
read:workflow-scheme:jira
read:workflow:jira
read:field-configuration:jira
read:issue-type-hierarchy:jira
read:webhook:jira
```

12. Navigieren Sie im linken Menü zu Authentifizierung und wählen Sie Hinzufügen.
13. Geben Sie eine Callback-URL wie https://us-east-1.console.aws.amazon.com/gluestudio/ oauth ein
14. Navigiere im linken Menü zu Einstellungen und scrolle nach unten zu den Authentifizierungsdetails. Notieren Sie sich die Client-ID und das Geheimnis.

Konfiguration von Jira Cloud-Verbindungen

Jira Cloud unterstützt den Grant-Typ AUTHORIZATION_CODE für. OAuth2

- Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Jira Cloud weiter, wo er sich anmelden und die angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Jira Cloud-Instanz gewähren muss.

- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Jira Cloud zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Jira Cloud weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Grant-Typ führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Eine öffentliche Jira Cloud-Dokumentation zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow findest du unter Enabling OAuth 2.0 (3LO).

So konfigurierst du eine Jira Cloud-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für die Verbindung in erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Jira Cloud aus.
   b. Stellen Sie die Jira Cloud-Umgebung bereit.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Jira Cloud-Entitäten lesen

Voraussetzung

Ein Jira Cloud-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Audit Record oder Issue. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Prüfprotokoll	Ja	Ja	Nein	Ja	Ja
Problem	Ja	Ja	Nein	Ja	Ja
Feld „Problem“	Nein	Nein	Nein	Ja	Nein
Konfiguration des Problemfeldes	Ja	Ja	Nein	Ja	Ja
Art des Problemlinks	Nein	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Schema zur Meldung von Problemen	Ja	Ja	Nein	Ja	Ja
Sicherheitsschema ausstellen	Nein	Nein	Nein	Ja	Nein
Art des Problems: Schema	Ja	Ja	Ja	Ja	Ja
Art des Problems: Bildschirmschema	Ja	Ja	Ja	Ja	Ja
Art des Problems	Nein	Nein	Nein	Ja	Nein
Jira-Einstellung	Ja	Nein	Nein	Ja	Nein
Jira-Einstellung für Fortgeschrittene	Nein	Nein	Nein	Ja	Nein
Jira-Einstellung global	Nein	Nein	Nein	Ja	Nein
Label (Bezeichnung)	Nein	Nein	Nein	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Ich selbst	Ja	Nein	Nein	Ja	Nein
Berechtigung	Nein	Nein	Nein	Ja	Nein.
Projekt	Ja	Ja	Ja	Ja	Ja
Kategorie des Projekts	Nein	Nein	Nein	Ja	Nein
Projekttyp	Nein	Nein	Nein	Ja	Nein
Informationen zum Server	Nein	Nein	Nein	Ja	Nein
Benutzer	Nein	Nein	Nein.	Ja	Nein
Workflow	Ja	Ja	Ja	Ja	Ja
Workflow-Schema	Nein	Ja	Nein	Ja	Ja
Zuordnung des Workflow-Schemas zum Projekt	Ja	Nein	Nein	Ja	Nein
Status des Workflows	Nein	Nein	Nein	Ja	Nein
Kategorie Workflow-Status	Nein	Nein	Nein	Ja	Nein

Beispiel:

```
jiracloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="JiraCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "audit-record",
        "API_VERSION": "v2"
    }
```

Details zur Jira Cloud-Entität und zum Feld:

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
Prüfprotokoll	Filter	String	"="
	from	DateTime	"="
	to	DateTime	"="
	id	Ganzzahl	N/A
	Übersicht	String	N/A
	Remote-Adresse	String	N/A
	authorAccountId	String	N/A
	created	String	N/A
	Kategorie	String	N/A
	eventSource	String	N/A
	description	String	N/A
	ObjektItem	Struct	N/A
	Geänderte Werte	Auflisten	N/A
	Zugeordnete Artikel	Auflisten	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
Gruppen	groupName	Auflisten	"="
	Name	String	N/A
	groupId	String	"="
Problem	Betroffene Version	String	"=, !="
	Rechtsnachfolger(in)	String	"=, !="
	Kategorie	String	"=, !="
	Komponente	String	"=, !="
	Schöpfer	String	"=, !="
	fällig	DateTime	N/A
	epic_link	String	"=, !="
	Filter	String	"=, !="
	Version korrigieren	String	"=, !="
	Hierarchieebene	Ganzzahl	"=, !="
	Problemschlüssel	String	"=, !=, >, <, >=, <="
	Link zum Problem	String	"=, !="
	issueLinkType	String	"=, !="
	labels	String	"=, !="
	Zuletzt angesehen	DateTime	„=, >, <, >=, <=, zwischen“
	level	String	"=, !="

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	übergeordneter	String	"=, !="
	priority	String	"=, !="
	project	String	"=, !="
	Typ des Projekts	String	"=, !="
	Berichterstatter	String	"=, !="
	Auflösung	String	"=, !="
	gelöst	DateTime	„=, >, <, >=, <=, zwischen“
	Sprint	String	"=, !="
	Status	String	"=, !="
	Typ	String	"=, !="
	Aktualisiert	DateTime	„=, >, <, >=, <=, zwischen“
	Wähler	String	"=, !="
	Wahlen	Ganzzahl	„=,! =, <, >, <=, >=, zwischen“
	Beobachter	String	"=, !="
	Beobachter	Ganzzahl	„=,! =, <, >, <=, >=, zwischen“
	Arbeitsverhältnis	Ganzzahl	„=,! =, <, >, <=, >=, zwischen“
	Abfrage validieren	String	"="

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	erweitern	String	"="
	fieldByKeys	Boolesch	"="
	id	String	N/A
	self	String	N/A
	Schlüssel	String	N/A
	Gerenderte Felder	Struct	N/A
	Eigenschaften	Auflisten	"="
	Namen	Struct	N/A
	schema	Struct	N/A
	Übergänge	Auflisten	N/A
	Operationen	Struct	N/A
	bearbeite Meta	Struct	N/A
	Changelog	Struct	N/A
	Versionierte Repräsentationen	Struct	N/A
	fields	Auflisten	"="
	fieldsToInclude	Struct	N/A
	Warnmeldungen	Auflisten	N/A
	created	DateTime	N/A
	Datum des Arbeitsprotokolls	DateTime	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
IssueEvents	id	Ganzzahl	N/A
	Name	String	N/A
Felder für Probleme	id	String	N/A
	Schlüssel	String	N/A
	Name	String	N/A
	benutzerdefiniert	Boolesch	N/A
	bestellbar	Boolesch	N/A
	navigierbar	Boolesch	N/A
	durchsuchbar	Boolesch	N/A
	Namen der Klauseln	Auflisten	N/A
	scope	Struct	N/A
	schema	Struct	N/A
Geben Sie Feldkonfigurationen ein	ist Standard	Boolesch	"="
	query	String	"="
	id	Ganzzahl	"="
	Name	String	N/A
	description	String	N/A
Art des Problemlinks	id	String	N/A
	Name	String	N/A
	nach innen	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	äußerlich	String	N/A
	self	String	N/A
Meldesysteme ausstellen	erweitern	String	"="
	self	String	N/A
	id	Ganzzahl	N/A
	Name	String	N/A
	description	String	N/A
	notificationScheme Events	Auflisten	N/A
	scope	Struct	N/A
Priorität der Ausgabe	self	String	N/A
	Status/Farbe	String	N/A
	description	String	N/A
	URL des Symbols	String	N/A
	Name	String	N/A
	id	String	N/A
	ist Standard	Boolesch	N/A
Lösungen für Probleme	self	String	N/A
	id	String	N/A
	description	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	Name	String	N/A
Sicherheitsschema ausstellen	self	String	N/A
	id	Ganzzahl	N/A
	Name	String	N/A
	description	String	N/A
	defaultSecurityLevelID	Ganzzahl	N/A
	Niveaus	Auflisten	N/A
Art des Problems	self	String	N/A
	id	String	N/A
	description	String	N/A
	URL des Symbols	String	N/A
	Name	String	N/A
	Unteraufgabe	Boolesch	N/A
	Avatar-ID	Ganzzahl	N/A
	entityId	String	N/A
	Hierarchieebene	Ganzzahl	N/A
	scope	Struct	N/A
Art des Problems: Schema	orderBy	String	"="
	erweitern	String	"="

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	queryString	String	"="
	id	String	N/A
	Name	String	N/A
	description	String	N/A
	defaultIssueTypeId	String	N/A
	ist Standard	Boolesch	N/A
Art des Problems: Bildschirmschema	queryString	String	"="
	orderBy	String	"="
	erweitern	String	"="
	id	String	"="
	Name	String	N/A
	description	String	N/A
Jira-Einstellungen	Schlüssel	String	N/A
	KeyFilter	String	"="
	id	String	N/A
	Wert	String	N/A
	Name	String	N/A
	desc	String	N/A
	Typ	String	N/A
	defaultValue	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	Beispiel	String	N/A
	allowedValues	Auflisten	N/A
Erweiterte Jira-Einstellungen	id	String	N/A
	Schlüssel	String	N/A
	Wert	String	N/A
	Name	String	N/A
	desc	String	N/A
	Typ	String	N/A
	defaultValue	String	N/A
	Beispiel	String	N/A
	allowedValues	Auflisten	N/A
Globale Jira-Einstellungen	Abstimmung aktiviert	Boolesch	N/A
	Ansehen aktiviert	Boolesch	N/A
	unassignedIssuesAllowed	Boolesch	N/A
	subTasksEnabled	Boolesch	N/A
	issueLinkingEnabled	Boolesch	N/A
	timeTrackingEnabled	Boolesch	N/A
	Dateianhänge aktiviert	Boolesch	N/A
	timeTrackingConfiguration	Struct	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
Label (Bezeichnung)	values	Auflisten	N/A
Ich selbst	erweitern	String	"="
	self	String	N/A
	accountId	String	N/A
	Art des Kontos	String	N/A
	E-Mail-Adresse	String	N/A
	Avatar-URLs	String	N/A
	displayName	String	N/A
	aktiv	Boolesch	N/A
	Zeitzone	String	N/A
	locale	String	N/A
	Gruppen	Struct	N/A
	Rollen in der Anwendung	Struct	N/A
Berechtigung	id	String	N/A
	Schlüssel	String	N/A
	Name	String	N/A
	Typ	String	N/A
	description	String	N/A
	Haben Sie die Erlaubnis	Boolesch	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	Veralteter Schlüssel	Boolesch	N/A
Projekt	orderBy	String	"="
	keys	Auflisten	"="
	query	String	"="
	Geben Sie Key ein	String	"="
	Kategorie-ID	Ganzzahl	"="
	action	String	"="
	erweitern	String	"="
	Status	Auflisten	"="
	self	String	N/A
	id	Ganzzahl	"="
	Schlüssel	String	N/A
	description	String	N/A
	lead	Struct	N/A
	Komponenten	Auflisten	N/A
	Arten von Problemen	Auflisten	N/A
	URL	String	N/A
	email	String	N/A
	Typ des zugewiesenen Mitarbeiters	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	versions	Auflisten	N/A
	Name	String	N/A
	Rollen	Struct	N/A
	Avatar-URLs	Struct	N/A
	Kategorie des Projekts	Struct	N/A
	projectTypeKey	String	N/A
	simplified	Boolesch	N/A
	Stil	String	N/A
	Lieblings-	Boolesch	N/A
	ist privat	Boolesch	N/A
	issueTypeHierarchy	Struct	N/A
	permissions	Struct	N/A
	Eigenschaften	Auflisten	"="
	uuid	String	N/A
	Einsicht	Struct	N/A
	deleted	Boolesch	N/A
	retentionTillDate	String	N/A
	Gelöschtes Datum	String	N/A
	Gelöscht von	Struct	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	archived	Boolesch	N/A
	Archiviertes Datum	String	N/A
	Archiviert von	Struct	N/A
	landedPageInfo	Struct	N/A
Kategorie des Projekts	self	String	N/A
	id	String	N/A
	Name	String	N/A
	description	String	N/A
Projekttyp	Schlüssel	String	N/A
	Formatierter Schlüssel	String	N/A
	description	String	N/A
	Beschreibung I18N Key	String	N/A
	icon	String	N/A
	color	String	N/A
Informationen zum Server	Basis-URL	String	N/A
	version	String	N/A
	Versionsnummern	Auflisten	N/A
	deploymentType	String	N/A
	Build-Nummer	Ganzzahl	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	Erstellungsdatum	DateTime	N/A
	Serverzeit	DateTime	N/A
	SCM-Informationen	String	N/A
	Server-Titel	String	N/A
	Gesundheitschecks	Auflisten	N/A
Benutzer	self	String	N/A
	accountId	String	N/A
	Art des Kontos	String	N/A
	E-Mail-Adresse	String	N/A
	Avatar-URLs	Struct	N/A
	displayName	String	N/A
	aktiv	Boolesch	N/A
	Zeitzone	String	N/A
	locale	String	N/A
	Gruppen	Struct	N/A
	Rollen in der Anwendung	Struct	N/A
	erweitern	String	N/A
Workflow	Workflow-Name	String	"="
	erweitern	String	"="

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	queryString	String	"="
	orderBy	String	"="
	ist aktiv	Boolesch	"="
	id	Struct	N/A
	description	String	N/A
	Übergänge	Auflisten	N/A
	Status	Auflisten	N/A
	ist Standard	Boolesch	N/A
	Schemata	Auflisten	N/A
	Projekte	Auflisten	N/A
	hasDraftWorkflow	Boolesch	N/A
	Operationen	Struct	N/A
	created	String	N/A
	Aktualisiert	String	N/A
Workflow-Schema	self	String	N/A
	id	Ganzzahl	N/A
	Name	String	N/A
	description	String	N/A
	Standard-Arbeitsablauf	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	issueTypeMappings	Struct	N/A
	originalDefaultWorkflow	String	N/A
	originalIssueTypeZuordnungen	Struct	N/A
	Einziehung	Boolesch	N/A
	lastModifiedUser	Struct	N/A
	Zuletzt geändert	String	N/A
	updateDraftIfBenötigt	Boolesch	N/A
	Arten von Problemen	Struct	N/A
Verknüpfung des Projekts Workflow-Schema	projectId	Ganzzahl	"="
	Projekt-IDs	Auflisten	N/A
	Workflow-Schema	Struct	N/A
Status des Arbeitsablaufs	self	String	N/A
	description	String	N/A
	URL des Symbols	String	N/A
	Name	String	N/A
	id	String	N/A
	StatusCategory	Struct	N/A
Kategorie Workflow-Status	self	String	N/A
	id	String	N/A

Object	Feld	Datentyp	Filter-Operatoren werden unterstützt
	Schlüssel	String	N/A
	Farbname	String	N/A
	Name	String	N/A

Abfragen partitionieren

Sie können die zusätzliche Spark-Option bereitstellen, `NUM_PARTITIONS` wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `NUM_PARTITIONS`: die Anzahl der Partitionen.

Beispiel:

```
jiraCloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="JiraCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "issue",
        "API_VERSION": "v2",
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Jira Cloud

Im Folgenden sind die Verbindungsoptionen für Jira Cloud aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Jira Cloud.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Jira Cloud Rest API-Version, die Sie verwenden möchten. Zum Beispiel: v2.
- `DOMAIN_URL`(String) — (Erforderlich) Die Jira Cloud ID, die Sie verwenden möchten.

- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `NUM_PARTITIONS`(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen und Hinweise für Jira Cloud Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Jira Cloud Connector:

- Der `Contains` Operator arbeitet nicht mit dem `resourceName` Feld, das vom Datentyp String ist.

Verbindung zu Kustomer herstellen

Kustomer ist eine leistungsstarke Plattform für Kundenerlebnisse, die alles, was Sie benötigen, um Ihre Kunden besser bedienen zu können, in einem easy-to-use Tool vereint.

Themen

- AWS Glue Unterstützung für Kustomer
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Kustomer
- Konfiguration von Kundenverbindungen
- Aus Kundenentitäten lesen
- Verbindungsoptionen für Kunden
- Einschränkungen für Kunden

AWS Glue Unterstützung für Kustomer

AWS Glue unterstützt Kustomer wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Kustomer abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Kustomer-API-Versionen

Die folgenden Kustomer-API-Versionen werden unterstützt:

- v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Kustomer

Bevor Sie Daten von Kustomer AWS Glue an unterstützte Ziele übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Konto bei Kustomer, das die Daten enthält, die Sie übertragen möchten.
- In den Einstellungen für Ihr Konto haben Sie einen API-Schlüssel erstellt. Weitere Informationen finden Sie unter Einen API-Schlüssel erstellen.
- Sie geben den API-Schlüssel an, AWS Glue während Sie die Verbindung herstellen.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Kundenkonto herstellen.

Einen API-Schlüssel erstellen

Um einen API-Schlüssel zu erstellen, mit dem Sie eine Verbindung für den Kustomer-Connector in AWS Glue Studio herstellen, gehen Sie wie folgt vor:

1. Melden Sie sich mit Ihren Zugangsdaten beim Kustomer-Dashboard an.
2. Wählen Sie im linken Menü das Einstellungssymbol.
3. Erweitern Sie das Drop-down-Menü „Sicherheit“ und wählen Sie API-Schlüssel aus.

4. Wählen Sie auf der Seite zur Erstellung des API-Schlüssels in der oberen rechten Ecke die Option API-Schlüssel hinzufügen aus.
5. Füllen Sie die obligatorischen Eingaben für den API-Schlüssel aus, der erstellt wird.
   - Name: beliebiger Name für Ihren API-Schlüssel.
   - Rollen: „Organisation“ muss ausgewählt werden, damit der Kunde APIs funktioniert.
   - Läuft ab (in Tagen): Die Anzahl der Tage, für die der API-Schlüssel gültig sein soll. Sie können den Wert „Läuft nie ab“ beibehalten, wenn es für Ihren Anwendungsfall geeignet ist.
6. Wählen Sie Create (Erstellen) aus.
7. Speichern Sie den Wert des API-Schlüssels (Token) zur weiteren Verwendung, um eine Verbindung für den Kustomer-Connector in AWS Glue Studio herzustellen.

Konfiguration von Kundenverbindungen

Um eine Kustomer-Verbindung zu konfigurieren:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret die verbundene App Consumer Secret mit `apiKey` als Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Verbindungen die Option Verbindung erstellen.
   b. Wählen Sie bei der Auswahl einer Datenquelle Kustomer aus.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
```

```
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Kundenentitäten lesen

Voraussetzung

Ein Kustomer-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Brands oder Cards. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Marken	Nein	Ja	Nein	Ja	Nein
Karten	Nein	Ja	Nein	Ja	Nein
Chat-Einstellungen	Nein	Nein	Nein	Ja	Nein
Unternehmen	Ja	Ja	Ja	Ja	Ja
Konversationen	Ja	Ja	Ja	Ja	Ja
Kunden	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Suchanfragen von Kunden gepinnt	Nein	Ja	Nein	Ja	Nein
Kunde sucht Position	Nein	Nein	Nein	Ja	Nein
E-Mail-Hooks	Nein	Ja	Nein	Ja	Nein
Web-Hooks	Nein	Ja	Nein	Ja	Nein
KB-Artikel	Nein	Ja	Nein	Ja	Nein
KB-Kategorien	Nein	Ja	Nein	Ja	Nein
KB-Formulare	Nein	Ja	Nein	Ja	Nein
KB-Routen	Nein	Ja	Nein	Ja	Nein
KB-Tags	Nein	Ja	Nein	Ja	Nein
KB-Vorlagen	Nein	Ja	Nein	Ja	Nein
KB-Themen	Nein	Ja	Nein	Ja	Nein
Klassen	Nein	Ja	Nein	Ja	Nein
KViews	Nein	Ja	Nein	Ja	Nein
Nachrichten	Ja	Ja	Ja	Ja	Ja
Hinweise	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Benachrichtigungen	Nein	Ja	Nein	Ja	Nein

Beispiel:

```
Kustomer_read = glueContext.create_dynamic_frame.from_options(
    connection_type="kustomer",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "brands",
        "API_VERSION": "v1"
    }
```

Angaben zur Entität und zum Feld des Kunden

Weitere Informationen zu den Entitäten und Felddetails finden Sie unter:

- Marken
- Karten
- Chat-Einstellungen
- Unternehmen
- Konversationen
- Kunden
- Suchanfragen von Kunden gepinnt
- Kunde sucht nach Stellen
- Hooks E-Mail
- Hooks Web
- KB-Artikel
- KB-Kategorien
- KB-Formulare
- KB-Routen

- KB-Tags
- KB-Vorlagen
- KB-Themen
- Klassen
- Ansichten
- Nachrichten
- Hinweise
- Benachrichtigungen

Kunden-API v1

Entität	Feld	Datentyp	Unterstützte Operatoren
Marken	id	String	N/A
	Name	String	N/A
	Icon-URL	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
	default	Boolesch	N/A
Karten	id	String	N/A
	Name	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	description	String	N/A
	URL	String	N/A
	Kontexte	Auflisten	N/A
Chat-Einstellungen	id	String	N/A
	EinstellungenVersion	Ganzzahl	N/A
	Widget-Typ	String	N/A
	version	Ganzzahl	N/A
	Name des Teams	String	N/A
	Begrüßung	String	N/A
	automatische Antwort	String	N/A
	embedIconUrl	String	N/A
	embedIconColor	String	N/A
	fallbackEmailSubject	String	N/A
	fallbackEmailIntroduction	String	N/A
	aktiviert	Boolesch	N/A
	outboundChatEnabled	Boolesch	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
	Nachricht außerhalb der Geschäftszeiten	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	offhoursImageUrl	String	N/A
	Schließbarer Chat	Boolesch	N/A
	Keine Historie	Boolesch	N/A
	Anlagen deaktivieren	Boolesch	N/A
	Lautstärkeregler	Struct	N/A
	singleSessonChat	Boolesch	N/A
	showTypingIndicatorNetz	Boolesch	N/A
Unternehmen	id	String	N/A
	Name	String	=,! =, ENTHÄLT
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiziertAT	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	tags	Auflisten	N/A
	domains	Auflisten	N/A
	E-Mails	Auflisten	N/A
	Telefone	Auflisten	N/A
	WhatsApps	Auflisten	N/A
	soziale	Auflisten	N/A
	URLs	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Standorte	Auflisten	N/A
	roleGroupVersions	Auflisten	N/A
	rev	Ganzzahl	N/A
Konversationen	id	String	N/A
	Name	String	=,! =, ENTHÄLT
	Vorschau	String	N/A
	Kanäle	Auflisten	N/A
	Status	String	=,! =, ENTHÄLT
	Anzahl der Nachrichten	Ganzzahl	=, !=, >, >=, <, <=
	Anzahl Notizen	Ganzzahl	=, !=, >, >=, <, <=
	Befriedigung	Ganzzahl	=, !=, >, >=, <, <=
	Grad der Zufriedenheit	Struct	N/A
	createdAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	updatedAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	Geändert am	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	lastActivityAt	DateTime	N/A
	Spam	Boolesch	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	endete	Boolesch	=, !=
	Endete am	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	Grund beendet	String	CONTAINS
	endedByType	String	N/A
	Importiert unter	String	N/A
	tags	Auflisten	N/A
	Vorgeschlagene Schlagworte	Auflisten	N/A
	sentiment	String	N/A
	Voraussagen	Auflisten	N/A
	Vorgeschlagene Abkürzungen	Auflisten	N/A
	firstMessageIn	Struct	N/A
	firstMessageOut	Struct	N/A
	lastMessageIn	Struct	N/A
	lastMessageOut	Struct	N/A
	lastMessageAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	lastMessageUnrespondedZu	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	lastMessageUnrespondedToSinceLastDone	Struct	N/A
	Zugewiesene Benutzer	Auflisten	N/A
	Zugewiesene Teams	Auflisten	N/A
	Erste Antwort	Struct	N/A
	firstResponseSinceLastDone	Struct	N/A
	Letzte Antwort	Struct	N/A
	Erste erledigt	Struct	N/A
	Zuletzt erledigt	Struct	N/A
	Richtung	String	=,! =, ENTHÄLT
	lastMessageDirection	String	N/A
	outboundMessageCount	Ganzzahl	N/A
	inboundMessageCount	Ganzzahl	N/A
	rev	Ganzzahl	N/A
	priority	Ganzzahl	=, !=, >, >=, <, <=
	roleGroupVersions	Auflisten	N/A
	Greifen Sie auf Override zu	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Assistent	Struct	N/A
	Phase	String	N/A
	Fähigkeiten	Auflisten	N/A
	matchedTi meBasedRegeln	Auflisten	N/A
Kunden	id	String	N/A
	Name	String	=,! =, ENTHÄLT
	displayName	String	N/A
	Farbe anzeigen	String	N/A
	Symbol anzeigen	String	N/A
	Externe ID	String	=,! =, ENTHÄLT
	externe IDs	Auflisten	N/A
	sharedExternalIds	Auflisten	N/A
	E-Mails	Auflisten	N/A
	Geteilte E-Mails	Auflisten	N/A
	Telefone	Auflisten	N/A
	Geteilte Telefone	Auflisten	N/A
	WhatsApps	Auflisten	N/A
	Facebook-IDs	Auflisten	N/A
	Instagram-Kinder	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	soziale	Auflisten	N/A
	Geteilte soziale Netzwerke	Auflisten	N/A
	URLs	Auflisten	N/A
	Standorte	Auflisten	N/A
	Aktive Benutzer	Auflisten	N/A
	Beobachter	Auflisten	N/A
	Aktueller Standort	Struct	N/A
	locale	String	=,! =, ENTHÄLT
	Zeitzone	String	N/A
	gender	String	=,! =, ENTHÄLT
	createdAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	updatedAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	Geändert am	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	lastActivityAt	DateTime	N/A
	deleted	Boolesch	N/A
	Letzte Konversation	Struct	N/A
	Die Konversation zählt	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Vorschau	Struct	N/A
	tags	Auflisten	N/A
	Progressiver Status	String	=,! =, ENTHÄLT
	verified	Boolesch	N/A
	rev	Ganzzahl	N/A
	Aktuelle Artikel	Auflisten	N/A
	Standardlänge	String	=,! =, ENTHÄLT
	Zufriedenheitsgrad	Struct	N/A
	roleGroupVersions	Auflisten	N/A
	Greifen Sie auf Override zu	Auflisten	N/A
	Name der Firma	String	N/A
	firstName	String	N/A
	lastName	String	N/A
Suchanfragen von Kunden gepinnt	id	String	N/A
	search	String	N/A
	createdAt	DateTime	N/A
Kunde sucht nach Stellen	id	String	N/A
	Positionen	Auflisten	N/A
	Kinder	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
	rev	Ganzzahl	N/A
E-Mail-Hooks	id	String	N/A
	description	String	N/A
	debug	Boolesch	N/A
	email	String	N/A
	eventName	String	N/A
	Titel	String	N/A
	hash	String	N/A
	Schlüssel	String	N/A
	createdAt	DateTime	N/A
	Geändertes AT	DateTime	N/A
	updatedAt	DateTime	N/A
Web-Hooks	id	String	N/A
	description	String	N/A
	eventName	String	N/A
	hash	String	N/A
	URL	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	createdAt	DateTime	N/A
	Geändert am	DateTime	N/A
	updatedAt	DateTime	N/A
	Titel	String	N/A
	version	Ganzzahl	N/A
	debug	Boolesch	N/A
KB-Artikel	id	String	N/A
	hash	String	N/A
	Titel	String	N/A
	Quelle	String	N/A
	Status	String	N/A
	scope	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	deleted	Boolesch	N/A
	Gelöscht am	DateTime	N/A
	Geändert am	DateTime	N/A
	Veröffentlicht am	DateTime	N/A
	tags	Auflisten	N/A
	categories	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Wissensdatenbanken	Auflisten	N/A
	Meta-Titel	String	N/A
	Meta-Beschreibung	String	N/A
	Meta-Schlüsselwörter	Auflisten	N/A
	Lang-Versionen	Struct	N/A
	Die neuesten Langs	Struct	N/A
KB-Kategorien	id	String	N/A
	hash	String	N/A
	createdAt	DateTime	N/A
	Geändertes AT	DateTime	N/A
	updatedAt	DateTime	N/A
	veröffentlicht	Boolesch	N/A
	Positionen	Auflisten	N/A
	Kategorie: Positionen	Auflisten	N/A
	Root	Boolesch	N/A
	lang	Struct	N/A
KB-Formulare	id	String	N/A
	Name	String	N/A
	Schnecke	String	N/A
	hash	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	body	String	N/A
	Einteilung	Auflisten	N/A
	Grundriss V2	Auflisten	N/A
	Komponenten V2	Struct	N/A
	Bedingungen	Struct	N/A
	fortgeschrittene	Boolesch	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	veröffentlicht bei	DateTime	N/A
	Geändert am	String	N/A
	veröffentlicht	Boolesch	N/A
	Auszüge	Auflisten	N/A
	Recaptcha	Boolesch	N/A
	Klasse	String	N/A
	channel	String	N/A
	Durchbiegung	Boolesch	N/A
	formHookEnabled	Boolesch	N/A
	Antwort von	String	N/A
	wcag	Boolesch	N/A
KB-Routen	id	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	URL	String	N/A
	Routing-Typ	String	N/A
	Routingfähige ID	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
KB-Tags	id	String	N/A
	Name	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändertes AT	DateTime	N/A
KB-Vorlagen	id	String	N/A
	Titel	String	N/A
	description	String	N/A
	Beta	Boolesch	N/A
	Manifest	Struct	N/A
	JSX-Snippets	Auflisten	N/A
	Bilder	Auflisten	N/A
	version	String	N/A
	createdAt	DateTime	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	updatedAt	DateTime	N/A
KB-Themen	id	String	N/A
	Name	String	N/A
	aktiv	Boolesch	N/A
	default	Boolesch	N/A
	lastfileUpdatedAt	DateTime	N/A
	benutzerdefiniert	Boolesch	N/A
	Status	String	N/A
	templateVersionId	String	N/A
	Titel der Vorlage	String	N/A
	Version der Vorlage	String	N/A
	Manifest	Struct	N/A
	Schnipsel konfiguri eren	Auflisten	N/A
	JSX-Snippets	Auflisten	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
	rev	Ganzzahl	N/A
Klassen	id	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Name	String	N/A
	icon	String	N/A
	color	String	N/A
	App deaktiviert	Boolesch	N/A
	Status	String	N/A
	updatedAt	DateTime	N/A
	createdAt	DateTime	N/A
	s3 DataUrl	String	N/A
KViews	id	String	N/A
	Ressource	String	N/A
	Vorlage	String	N/A
	context	String	N/A
	Meta	Struct	N/A
	App deaktiviert	Boolesch	N/A
	aktiviert	Boolesch	N/A
	fortgeschrittene	Boolesch	N/A
	Einteilung	Auflisten	N/A
	Komponenten	Struct	N/A
	Bedingungen	Struct	N/A
	rev	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	createdAt	DateTime	N/A
	Geändert am	DateTime	N/A
	updatedAt	DateTime	N/A
Benachrichtigungen	id	String	N/A
	Name	String	N/A
	Status	String	N/A
	event	Struct	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
Nachrichten	id	String	N/A
	Externe ID	String	N/A
	channel	String	=,! =, ENTHÄLT
	App	String	N/A
	size	Ganzzahl	=, !=, >, >=, <, <=
	Richtung	String	=,! =, ENTHÄLT
	Vorschau	String	N/A
	subject	String	N/A
	Meta	Struct	N/A
	Status	String	=,! =, ENTHÄLT
	Richtungstyp	String	=,! =, ENTHÄLT

Entität	Feld	Datentyp	Unterstützte Operatoren
	Zugewiesene Teams	Auflisten	N/A
	Zugewiesene Benutzer	Auflisten	N/A
	Fehler bei	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	auto	Boolesch	=, !=
	Gesendet	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	createdAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	updatedAt	DateTime	N/A
	Geändert am	DateTime	N/A
	redigiert	Boolesch	N/A
	createdByTeams	Auflisten	N/A
	rev	Ganzzahl	N/A
	Reaktionen	Auflisten	N/A
	Erkennungen von Absichten	Auflisten	N/A
Hinweise	id	String	N/A
	body	String	CONTAINS
	createdAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN

Entität	Feld	Datentyp	Unterstützte Operatoren
	updatedAt	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	Geändert am	DateTime	=,! =, >, >=, <, <=, ZWISCHEN
	createdByTeams	Auflisten	N/A

Abfragen partitionieren

Feldbasierte Partitionierung

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2023-01-15T11:18:39.205Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt:

Entity-Name	Partitionierung von Feldern	Datentyp
Unternehmen	Geändertes AT	DateTime
Konversationen	createdAt, updatedAt, GeändertAm, EndDatum, lastMessageAt	DateTime
	Anzahl Nachrichten, Anzahl Notizen	BigInteger
	priority	Ganzzahl
Kunden	createdAt, updatedAt, GeändertAm	DateTime
Nachrichten	ErrorAt, SenTat, CreatedAt	DateTime
	size	BigInteger
Hinweise	createdAt, updatedAt, GeändertAm	DateTime

Beispiel:

```
Kustomer_read = glueContext.create_dynamic_frame.from_options(
    connection_type="kustomer",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversation",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2023-01-15T11:18:39.205Z"
        "UPPER_BOUND": "2023-02-15T11:18:39.205Z"
        "NUM_PARTITIONS": "2"
    }
```

Verbindungsoptionen für Kunden

Im Folgenden sind die Verbindungsoptionen für Kustomer aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Kustomer.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. Kustomer Rest API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. URL der Kundeninstanz.

Einschränkungen für Kunden

Im Folgenden finden Sie Einschränkungen oder Hinweise für Kustomer:

- Die `Customer Searches` Entität wird nicht unterstützt, da in der Kustomer-API-Dokumentation kein Endpunkt für sie deklariert wurde.
- Die Unterstützung von Filterung und inkrementeller Übertragung auf der `Klasses` Entität wird nicht unterstützt.
- Order by kann für mehrere zutreffende Felder in einer einzigen Anfrage unterstützt werden.

  Es wurde jedoch beobachtet, dass sich die Reihenfolge nach Funktionalität in mehreren Feldern bei einigen Kombinationen von der SaaS-Seite aus inkonsistent verhält. Dies ist nicht vorhersehbar, da es „n“ Kombinationen geben kann, die zu falschen Sortierergebnissen führen könnten. Zum Beispiel:

Für die `Customers` Entität ergibt die Reihenfolge nach `progressiveStatus desc, name asc` nicht das richtige sortierte Ergebnis. Es sortiert nur anhand der `progressiveStatus` Reihenfolge. Wenn ein solches Verhalten beobachtet wird, können Sie ein einzelnes Feld verwenden, nach dem sortiert werden soll.

- Die Reihenfolge nach im Feld 'id' wird nur von den `Messages` Entitäten `Conversations` und als Abfrageparameter unterstützt. Zum Beispiel: https://api.kustomerapp.com/v1/ Konversationen?sort=desc (Sortiert die Ergebnisse nach 'id' in absteigender Reihenfolge.)

  Darüber hinaus wird jeder andere Filter oder jede Bestellung in einem anderen Feld in einen POST-Anforderungstext übersetzt, dessen API-Endpunkt POST https://api.kustomerapp.com/v1/customers/search lautet. Um die Unterstützung der Sortierung nach 'id' in `Conversations` und zu ermöglichen`Messages`, sollte entweder nur „Nach ID bestellen“ vorhanden sein oder ein anderer Filter und/oder Order by in einem anderen zutreffenden Feld.
- Kustomer ermöglicht das Abrufen von maximal 10.000 Datensätzen, unabhängig davon, ob es sich um eine gefilterte oder eine ungefilterte Anfrage handelt. Aufgrund dieser Einschränkung kann es bei jeder Entität, die mehr als 10.000 Datensätze besitzt, zu Datenverlust kommen. Es gibt zwei mögliche Problemumgehungen, mit denen Sie dieses Problem teilweise abmildern können:
  - Wenden Sie Filter an, um eine bestimmte Gruppe von Datensätzen abzurufen.
  - Wenn es mehr als 10.000 Datensätze gibt, auf die ein Filter angewendet wurde, wenden Sie einen aufeinanderfolgenden Filterwert in einer neuen nachfolgenden Anfrage an oder wenden Sie Bereiche in Filtern an. Zum Beispiel:

    Der Filterausdruck der ersten Anfrage: `modifiedAt >= 2022-03-15T05:26:23.000Z and modifiedAt < 2023-03-15T05:26:23.000Z`

    Gehen Sie davon aus, dass dadurch das Limit von 10.000 Datensätzen ausgeschöpft ist.

    Eine weitere Anfrage kann mit FilterExpression ausgelöst werden: `modifiedAt >= 2023-03-15T05:26:23.000Z`
- Als SaaS-Verhalten unterstützt der `CONTAINS` Operator in Kustomer den Abgleich nur bei vollständigen Wörtern und nicht bei teilweisen Übereinstimmungen innerhalb eines Worts. Beispiel: „body CONTAINS 'test record'“ entspricht einem Datensatz, bei dem das Wort 'test' im Feld 'body' steht. „body CONTAINS 'test'“ entspricht jedoch nicht einem Datensatz, der 'testAnotherRecord' im Feld 'body' enthält.

Verbindung herstellen zu LinkedIn

LinkedIn ist ein kostenpflichtiges Marketinginstrument, das über verschiedene gesponserte Beiträge und andere Methoden Zugang zu LinkedIn sozialen Netzwerken bietet. LinkedIn ist ein leistungsstarkes Marketinginstrument für B2B-Unternehmen, um Leads zu generieren, Online-Bekanntheit zu erlangen, Inhalte zu teilen und vieles mehr.

Themen

- AWS Glue Unterstützung für LinkedIn
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration LinkedIn
- LinkedIn Verbindungen konfigurieren
- Aus LinkedIn Entitäten lesen
- LinkedIn Verbindungsoptionen
- Ein LinkedIn Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für LinkedIn

AWS Glue unterstützt LinkedIn wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen LinkedIn.

Wird es als Ziel unterstützt?

Nein.

Unterstützte LinkedIn API-Versionen

202406 (Juni 2024)

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration LinkedIn

Bevor Sie das Programm AWS Glue für die Übertragung von verwenden können LinkedIn, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein LinkedIn Konto. Weitere Informationen zum Erstellen eines Kontos finden Sie unter LinkedInKonto erstellen.
- Ihr LinkedIn Konto ist für den API-Zugriff aktiviert.
- Sie haben eine `OAuth2 API` Integration in Ihrem LinkedIn Konto erstellt. Diese Integration stellt die Kundenanmeldedaten bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter the section called “Ein LinkedIn Konto erstellen”.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem LinkedIn Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun LinkedIn.

LinkedIn Verbindungen konfigurieren

LinkedIn unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für OAuth2.

Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, `OAuth` da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Benutzer können sich dafür entscheiden, ihre eigene verbundene App in zu erstellen und ihre eigene Client-ID LinkedIn und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin weitergeleitet, LinkedIn um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt sowohl zu einem Aktualisierungstoken als auch zu einem Zugriffstoken. Das Zugriffstoken läuft 60 Tage nach seiner Erstellung ab. Ein neues Zugriffstoken kann mit dem Aktualisierungstoken abgerufen werden.

Öffentliche LinkedIn Dokumentation zur Erstellung einer verbundenen App für `Authorization Code OAuth` Flow finden Sie unter Authorization Code Flow (3-beinig OAuth).

Eine Verbindung konfigurieren LinkedIn

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:

   - Für vom Kunden verwaltete verbundene App — Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

- Für AWS verwaltete verbundene Apps — Leeres Geheimnis oder Geheimnis mit einem temporären Wert.

> **Note**
>
> Es ist ein Muss, pro Verbindung in ein Geheimnis zu erstellen AWS Glue.

2. Stellen Sie im unter Datenverbindungen eine Verbindung her AWS Glue Studio, indem Sie die folgenden Schritte ausführen:

   1. Wählen Sie bei der Auswahl eines Verbindungstyps LinkedIn.
   2. Stellen Sie die LinkedIn Umgebung bereit.
   3. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   4. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.
   5. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus LinkedIn Entitäten lesen

Voraussetzungen

Ein LinkedIn Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Konten hinzufügen	Ja	Ja	Ja	Ja	Nein
Kampagnen	Ja	Ja	Ja	Ja	Nein
Kampagnengruppen	Ja	Ja	Ja	Ja	Nein
Kreative	Ja	Ja	Ja	Ja	Nein
Werbeanalytik	Ja	Nein	Nein	Ja	Nein
Anzeigenanalyse Alle AdAcocunts	Ja	Nein	Nein	Ja	Nein
Ad Analytics Alle Kampagnen	Ja	Nein	Nein	Ja	Nein
Werbeanalyse Alle CampaignGroups	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Ad Analytics Alle AdCreatives	Ja	Nein	Nein	Ja	Nein
Statistiken teilen	Ja	Nein	Nein	Ja	Nein
Seitenstatistiken	Ja	Nein	Nein	Ja	Nein
Follower-Statistik	Ja	Nein	Nein	Ja	Nein

Beispiel

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="linkedin",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "adaccounts",
        "API_VERSION": "202406"
    }
)
```

LinkedIn Entitäts- und Felddetails

Datentyp des Felds	Unterstützte Filteroperatoren
String	=
DateTime	ZWISCHEN, =
Numerischer Wert	=
Boolesch	=

LinkedIn Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für LinkedIn:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in. LinkedIn Zum Beispiel ADAccounts.
- `API_VERSION`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. LinkedIn Rest-API-Version, die Sie verwenden möchten. Der Wert wird 202406 sein, da LinkedIn derzeit nur Version 202406 unterstützt wird.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für die ausgewählte Entität auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Ein LinkedIn Konto erstellen

Eine LinkedIn App und OAuth Anmeldeinformationen erstellen

1. Navigieren Sie zu Ihrer LinkedIn Developer Network-Seite und melden Sie sich mit Ihren LinkedIn Kontoanmeldeinformationen an.
2. Navigieren Sie zur Seite „Meine Apps“ und wählen Sie „Anwendung erstellen“, um eine neue LinkedIn App zu erstellen.
3. Geben Sie die folgenden Details in das App-Registrierungsformular ein:
    - Firmenname — Wählen Sie ein bestehendes Unternehmen aus oder erstellen Sie ein neues Unternehmen.
    - Name — Geben Sie den Namen der Anwendung ein.
    - Beschreibung — Geben Sie die Beschreibung der Anwendung ein.
    - Anwendungslogo — Wählen Sie eine Bilddatei als Ihr Anwendungslogo aus.
    - Verwendung der Anwendung — Wählen Sie die Verwendung Ihrer Anwendung aus.
    - Website-URL — Geben Sie die URL der Website ein, die detaillierte Informationen zu Ihrer Anwendung enthält.
    - Geschäftliche E-Mail-Adresse — Geben Sie Ihre geschäftliche E-Mail-Adresse ein.

- Geschäftstelefon — Geben Sie Ihre geschäftliche Telefonnummer ein.
- LinkedIn API-Nutzungsbedingungen — Lesen Sie und stimmen Sie zu.

4. Wählen Sie nach dem Ausfüllen des App-Registrierungsformulars „Senden“.

   Sie werden zur Authentifizierungsseite weitergeleitet, auf der die Authentifizierungsschlüssel (Client-ID und Client Secret) und andere relevante Details angezeigt werden.

5. Wenn Ihre Webanwendung Zugriff auf die E-Mail-Adresse des Benutzers von seinem LinkedIn Konto aus benötigt, wählen Sie die `r_emailaddress` Berechtigung aus. Darüber hinaus können Sie eine autorisierte Weiterleitung URLs für Ihre LinkedIn Anwendung angeben.

Eine Seite im LinkedIn Konto erstellen

1. Navigieren Sie zu LinkedIn Developer Products.
2. Wählen Sie in der oberen rechten Ecke der Seite mit den LinkedIn Entwicklerprodukten die Option Meine Apps aus.
3. Wählen Sie in der oberen rechten Ecke der Seite Meine Apps die Option App erstellen aus.
4. Geben Sie auf der Seite App erstellen Ihren App-Namen in das Feld App-Name ein.
5. Geben Sie im Feld LinkedIn Seite den Namen oder die URL Ihrer Unternehmensseite ein.

   > **Note**
   >
   > Wenn Sie noch keine LinkedIn Seite haben, können Sie eine erstellen, indem Sie Neue erstellen auswählen LinkedIn.

6. Geben Sie im Feld Datenschutzrichtlinien-URL Ihre Datenschutzrichtlinie-URL ein.
7. Wählen Sie Logo hochladen, um ein Bild hochzuladen, das Benutzern angezeigt werden soll, wenn sie sich mit Ihrer App autorisieren.
8. Wählen Sie im Abschnitt Rechtliche Vereinbarung die Option Ich habe diese Bedingungen gelesen und stimme ihnen zu.
9. Wähle App erstellen.

   Ihre neue App wird erstellt und ist auf dem Tab Meine Apps verfügbar.

Veröffentlichung von Kampagnenanzeigen in LinkedIn

1. Melden Sie sich bei Campaign Manager an.
2. Wählen Sie eine bestehende Kampagnengruppe aus oder wählen Sie Erstellen, um eine neue zu erstellen.
3. Wählen Sie Ihr Ziel aus.
4. Wählen Sie Ihre Gruppe, Ihr Budget und Ihren Zeitplan aus.
5. Bauen Sie Ihre Zielgruppe auf.
6. Wählen Sie Ihr Anzeigenformat aus.
7. Wählen Sie Ihr Budget und Ihren Zeitplan aus.
8. Richten Sie Ihre Anzeige (n) ein.
9. Überprüfen und starten.

Einschränkungen

Für die Analytics-Felder`ad_analytics_all_adAccounts`, `ad_analytics_all_campaignsad_analytics_all_campaign_groups`, und ist `ad_analytics_all_adCreatives` ein Filter erforderlich, um die Datensätze abzurufen.

Verbindung zu Mailchimp herstellen

Mailchimp ist eine all-in-one Marketingplattform, mit der Sie Ihre Kunden, Kunden und andere interessierte Parteien verwalten und mit ihnen sprechen können. Ihr Marketingansatz konzentriert sich auf gesunde Kontaktmanagementpraktiken, wunderschön gestaltete E-Mails, einzigartige automatisierte Workflows und leistungsstarke Datenanalysen. Wenn du ein Mailchimp-Nutzer bist, kannst du dich mit deinem Mailchimp-Konto AWS Glue verbinden. Anschließend können Sie Mailchimp als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Mailchimp und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Mailchimp
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Mailchimp konfigurieren
- Konfiguration von Mailchimp-Verbindungen
- Aus Mailchimp-Entitäten lesen

- Verbindungsoptionen für Mailchimp
- Ein Mailchimp-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Mailchimp

AWS Glue unterstützt Mailchimp wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Mailchimp abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Mailchimp-API-Versionen

3.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
```

```
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Mailchimp konfigurieren

Bevor du die Übertragung von Mailchimp verwenden AWS Glue kannst, musst du die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Mailchimp-Konto mit E-Mail und Passwort. Weitere Informationen zum Erstellen eines Kontos findest du unter Ein Mailchimp-Konto erstellen.
- Sie müssen AWS ein Konto mit dem Dienstzugriff auf erstellt haben. AWS Glue
- Stellen Sie sicher, dass Sie eine der folgenden Ressourcen erstellt haben. Diese Ressourcen enthalten Anmeldeinformationen, mit denen AWS Glue Sie sicher auf Ihre Daten zugreifen können, wenn Sie authentifizierte Anrufe an Ihr Konto tätigen:
  - Eine Entwickler-App, die OAuth 2.0-Authentifizierung unterstützt. Weitere Informationen zum Erstellen einer Entwickler-App findest du unter Ein Mailchimp-Konto erstellen.

Wenn du diese Anforderungen erfüllst, bist du bereit, dich mit deinem AWS Glue Mailchimp-Konto zu verbinden. Für typische Verbindungen müssen Sie in Mailchimp nichts weiter tun.

Konfiguration von Mailchimp-Verbindungen

Mailchimp unterstützt die folgenden zwei Arten von Authentifizierungsmechanismen:

- Mailchimp unterstützt den Grant-Typ. `AUTHORIZATION_CODE`
  - Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, `OAuth` da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App verlassen, für die er außer seiner Mailchimp-Client-ID und seinem Client Secret keine `OAuth` zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Mailchimp weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Mailchimp-Instanz gewähren muss.
  - Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Mailchimp zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Mailchimp weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen zu autorisieren AWS Glue .
  - Die öffentliche Mailchimp-Dokumentation zur Erstellung einer verbundenen App für `AUTHORIZATION_CODE OAuth` Flow findest du unter Zugreifen auf Daten im Namen anderer Nutzer mit 2. OAuth
- Benutzerdefinierte Authentifizierung — Die öffentliche Mailchimp-Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie unter Über API-Schlüssel.

Um eine Mailchimp-Verbindung zu konfigurieren:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - `OAuth`hauth — Für vom Kunden verwaltete verbundene App: Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

- Benutzerdefinierte Authentifizierung — Für vom Kunden verwaltete verbundene Apps: Secret sollte das Consumer Secret der verbundenen App mit „api_key“ als Schlüssel enthalten.

> **Note**
>
> Es ist ein Muss, für jede Verbindung in ein Secret zu erstellen. AWS Glue

2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Verbindungen die Option Verbindung erstellen aus.
   b. Wählen Sie bei der Auswahl einer Datenquelle Mailchimp aus.
   c. Stellen Sie den Mailchimp bereit. `instanceUrl`
   d. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
    "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "secretsmanager:DescribeSecret",
                    "secretsmanager:GetSecretValue",
                    "secretsmanager:PutSecretValue",
                    "ec2:CreateNetworkInterface",
                    "ec2:DescribeNetworkInterface",
                    "ec2:DeleteNetworkInterface",
                ],
                "Resource": "*"
            }
        ]
    }
```

   e. Wählen Sie den Authentifizierungstyp aus, um eine Verbindung zu Mailchimp herzustellen:
      - Für `OAuth` die Authentifizierung — Geben Sie die Token-URL und die benutzerverwaltete Client-Anwendung ClientId von Mailchimp an, zu der Sie eine Verbindung herstellen möchten.
      - Für benutzerdefinierte Authentifizierung — Wählen Sie den Authentifizierungstyp CUSTOM aus, um eine Verbindung zu Mailchimp herzustellen.

f. Wählen Sie den aus`secretName`, den Sie für diese Verbindung verwenden möchten, AWS Glue um die Token zu platzieren.

g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Mailchimp-Entitäten lesen

Voraussetzungen

Ein Mailchimp-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

- Missbrauchsberichte
- Automation
- Kampagnen
- Klicken Sie auf Details
- Listen
- Mitglieder
- Details öffnen
- Segmente
- Geschäfte
- Abgemeldet

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Automatisierung	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Kampagnen	Nein	Nein	Nein	Nein	Nein
Listen	Ja	Ja	Nein	Ja	Ja
Meldet Missbrauch	Nein	Ja	Nein	Ja	Ja
Berichte werden geöffnet	Nein	Ja	Nein	Ja	Ja
Berichte Klicken Sie hier	Ja	Ja	Nein	Ja	Ja
Berichte abbestellen	Nein	Ja	Nein	Ja	Ja
Segment	Nein	Ja	Nein	Ja	Ja
Mitglieder segmentieren	Ja	Ja	Nein	Ja	Nein
Geschäfte	Ja	Ja	Ja	Ja	Nein

Beispiel

```
mailchimp_read = glueContext.create_dynamic_frame.from_options(
            connection_type="mailchimp",
            connection_options={

                  "connectionName": "connectionName",

                  "ENTITY_NAME": "stores",
"INSTANCE_URL": "https://us14.api.mailchimp.com",
                  "API_VERSION": "3.0"
```

```
                })
```

Mailchimp-Entitäts- und Felddetails

- Missbrauchsberichte
- Automation
- Kampagnen
- Klicken Sie auf Details
- Listen
- Mitglieder
- Details öffnen
- Segmente
- Geschäfte
- Abgemeldet

Abfragen partitionieren

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2024-07-01T00:00:00.000Z"
  ```
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

In der folgenden Tabelle werden die Informationen zur Unterstützung von Feldern zur Partitionierung von Entitäten beschrieben:

Entity-Name	Partitionierung von Feldern	Datentyp

Beispiel:

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mailchimp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "automations",
        "API_VERSION": "3.0",
        "INSTANCE_URL": "https://us14.api.mailchimp.com",
        "PARTITION_FIELD": "create_time",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Verbindungsoptionen für Mailchimp

Im Folgenden sind die Verbindungsoptionen für Mailchimp aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Mailchimp.
- `INSTANCE_URL`(String) — (Erforderlich) Eine gültige Mailchimp-Instanz-URL.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. Mailchimp Engage Rest API-Version, die Sie verwenden möchten. Zum Beispiel: 3.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.

- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Ein Mailchimp-Konto erstellen

1. Navigiere zur Mailchimp-Anmeldeseite, gib deine E-Mail-ID und dein Passwort ein und wähle dann Sign up.
2. Öffne die Bestätigungs-E-Mail von Mailchimp und wähle den Bestätigungslink, um dein Konto zu verifizieren.

   > **Note**
   >
   > Die Zeit, die benötigt wird, um die Aktivierungs-E-Mail zu erhalten, kann variieren. Wenn Sie die Aktivierungs-E-Mail nicht erhalten haben, überprüfen Sie Ihren Spam-Ordner und lesen Sie unsere Tipps zur Fehlerbehebung bei Aktivierungs-E-Mails. Mailchimp blockiert Anmeldungen von rollenbasierten E-Mail-Adressen wie admin@pottedplanter.com oder security@example.com.

   Wenn Sie sich zum ersten Mal in Ihr Konto einloggen, fragt Mailchimp nach den erforderlichen Informationen. Mailchimp verwendet diese Informationen, um sicherzustellen, dass Ihr Konto den Nutzungsbedingungen entspricht, und um Ihnen Hinweise zu geben, die für Sie und die Bedürfnisse Ihres Unternehmens relevant sind.
3. Geben Sie Ihre Daten ein, folgen Sie den Anweisungen, um den Aktivierungsprozess abzuschließen, und legen Sie mit Ihrem neuen Mailchimp-Konto los.

Registrierung einer Anwendung **`OAuth2.0`**

1. Navigiere zur Mailchimp-Anmeldeseite, gib deine E-Mail-ID und dein Passwort ein und wähle Anmelden.
2. Wählen Sie das Benutzersymbol in der oberen rechten Ecke und wählen Sie dann Konto und Abrechnung aus dem Drop-down-Menü aus.
3. Wählen Sie Extras und dann Registrierte Apps aus dem Drop-down-Menü aus.
4. Suchen Sie nach einer App und wählen Sie „App registrieren“.
5. Geben Sie die folgenden Details ein:
   - App-Name — Name der App.
   - Firma/ Organisation — Name Ihres Unternehmens oder Ihrer Organisation.
   - App-Website — Website der App.
   - Umleitungs-URI — Ein Umleitungs-URI-Muster ist ein URI-Pfad (oder eine durch Kommas getrennte Liste von Pfaden), zu dem Mailchimp (falls angefordert) weiterleiten kann, wenn der Anmeldevorgang abgeschlossen ist. Beispiel: `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`
6. Wählen Sie Create (Erstellen) aus.
7. Die Client-ID und das Kundengeheimnis werden jetzt sichtbar sein. Kopieren Sie sie und speichern Sie sie an einem sicheren Ort. Wählen Sie dann „Fertig“.

> **Note**
>
> Ihre Client-ID- und Client Secret-Zeichenfolgen sind Anmeldeinformationen, die verwendet werden, um eine Verbindung mit diesem Connector herzustellen, wenn Sie AppFlow oder verwenden AWS Glue.

Generieren eines API-Schlüssels

1. Navigiere zur Mailchimp-Anmeldeseite, gib deine E-Mail-ID und dein Passwort ein und wähle Anmelden.
2. Wählen Sie das Benutzersymbol in der oberen rechten Ecke und wählen Sie dann Konto und Abrechnung aus dem Drop-down-Menü aus.
3. Wählen Sie Extras und dann API-Schlüssel aus dem Drop-down-Menü aus.

4. Wählen Sie Create A Key aus.
5. Geben Sie einen Namen für den Schlüssel ein und wählen Sie Schlüssel generieren.

   Auf der nächsten Seite wird der generierte API-Schlüssel angezeigt.
6. Kopieren Sie Ihren Schlüssel, speichern Sie ihn sicher und wählen Sie Fertig.

Einschränkungen

Die folgenden Einschränkungen gelten für den Mailchimp-Connector:

- Die Filterung wird nur von`Campaigns`,, `Automations` `Lists` `Open` `Details``Members`, und `Segments` Entitäten unterstützt.
- Wenn Sie einen Filter für ein `DateTime` Datentypfeld verwenden, müssen Sie Werte in diesem Format übergeben: `yyyy-mm-ddThh:MM:ssZ`

Verbindung zu Microsoft Dynamics 365 CRM herstellen

Microsoft Dynamics 365 ist eine Produktlinie intelligenter Geschäftsanwendungen für Enterprise Resource Planning und Customer Relationship Management.

Themen

- AWS Glue Unterstützung für Microsoft Dynamics 365
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Microsoft Dynamics 365 CRM
- Konfiguration von Microsoft Dynamics 365 CRM-Verbindungen
- Aus Microsoft Dynamics 365 CRM-Entitäten lesen
- Referenz zu Microsoft Dynamics 365 CRM-Verbindungsoptionen
- Einschränkungen

AWS Glue Unterstützung für Microsoft Dynamics 365

AWS Glue unterstützt Microsoft Dynamics 365 wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Microsoft Dynamics 365 abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Microsoft Dynamics 365 CRM API-Versionen

v9.2.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Microsoft Dynamics 365 CRM

Bevor Sie Daten aus Microsoft Dynamics 365 CRM übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

- Sie haben ein Microsoft Dynamics 365 CRM-Entwicklerkonto bei ClientId und Secret.
- Ihr Microsoft Dynamics 365 CRM-Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Microsoft Dynamics 365 CRM-Konto herstellen. Für typische Verbindungen müssen Sie in Microsoft Dynamics 365 CRM nichts weiter tun.

Konfiguration von Microsoft Dynamics 365 CRM-Verbindungen

AUTHORIZATION_CODE Art des Zuschusses

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Microsoft Dynamics 365 CRM weiter, wo der Benutzer sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Microsoft Dynamics 365 CRM-Instanz gewähren muss.
- Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Microsoft Dynamics 365 CRM zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Microsoft Dynamics 365 CRM weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Eine öffentliche Microsoft Dynamics 365 CRM-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter | Microsoft Learn. Registrierung der Microsoft-App.

Microsoft Dynamics 365 CRM unterstützt die OAuth2 2.0-Authentifizierung.

So konfigurieren Sie eine Microsoft Dynamics 365 CRM-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in einen geheimen Schlüssel zu erstellen AWS Glue.
   - Für den AuthorizationCode Zuschusstyp:

     Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Client Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Stellen AWS Glue Studio Sie unter Datenverbindungen eine Verbindung her, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Microsoft Dynamics 365 CRM aus.
   b. Wählen Sie die INSTANCE_URL der Microsoft Dynamics 365 CRM-Instanz aus, mit der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, die die folgenden AWS Glue Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie Token-URL und Autorisierungscode-URL aus, um auf Ihren Microsoft Dynamics 365 CRM-Arbeitsbereich zuzugreifen.

e. Stellen Sie die benutzerverwaltete Client-Anwendung ClientId Ihrer Microsoft Dynamics 365 CRM-App bereit.

f. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte. Wählen Sie Weiter aus.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Microsoft Dynamics 365 CRM-Entitäten lesen

Voraussetzungen

- Ein Microsoft Dynamics 365 CRM-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Kontakte oder Konten. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Dynamische Entität	Ja	Ja	Ja	Ja	Ja

Beispiel

```
dynamics365_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="microsoftdynamics365crm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dynamic_entity",
        "API_VERSION": "v9.2",
        "INSTANCE_URL": "https://{tenantID}.api.crm.dynamics.com"
    }
```

Microsoft Dynamics 365 CRM-Entitäts- und Felddetails

Entitäten mit dynamischen Metadaten:

Microsoft Dynamics 365 CRM bietet Endpunkte zum dynamischen Abrufen von Metadaten. Daher wird bei dynamischen Entitäten die Operatorunterstützung auf Datentypebene erfasst.

Entität	Datentyp	Unterstützte Operatoren
Dynamische Entität	DateTime	=, <, <=, >, >=, ZWISCHEN
	Datum	=, <, <=, >, >=
	String	=, !=
	Double	=, <, <=, >, >=
	Ganzzahl	=, <, <=, >, >=
	Dezimal	=, <, <=, >, >=
	Long	=, <, <=, >, >=
	BigInteger	=, <, <=, >, >=
	Auflisten	N/A
	Struct	N/A
	Zuordnung	N/A

Abfragen partitionieren

Microsoft Dynamics 365 CRM unterstützt nur feldbasierte Partitionierung.

Zusätzliche Spark-OptionenPARTITION_FIELD,LOWER_BOUND,UPPER_BOUND, NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für Datetime akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:. "2024-01-30T06:47:51.000Z"
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt:

Entity Name	Partitionierung von Feldern	DataType
Dynamische Entität (Standard entität)	Dynamische DateTime Felder, die abfragbar sind	erstellt am, geändert am
Dynamische Entität (benutzer definierte Entität)	erstellt am, geändert am	erstellt am, geändert am

Beispiel

```
dynamics365_read = glueContext.create_dynamic_frame.from_options(
    connection_type="microsoftdynamics365crm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dynamic_entity",
        "API_VERSION": "v9.2",
        "instanceUrl": "https://{tenantID}.api.crm.dynamics.com"
        "PARTITION_FIELD": "createdon"
        "LOWER_BOUND": "2024-01-30T06:47:51.000Z"
        "UPPER_BOUND": "2024-06-30T06:47:51.000Z"
        "NUM_PARTITIONS": "10"
```

```
    }
```

Referenz zu Microsoft Dynamics 365 CRM-Verbindungsoptionen

Im Folgenden sind Verbindungsoptionen für Microsoft Dynamics 365 CRM aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Microsoft Dynamics 365 CRM.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Microsoft Dynamics 365 CRM Rest API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `INSTANCE_URL`(Zeichenfolge) — (Erforderlich) Eine gültige Microsoft Dynamics 365 CRM-Instanz-URL mit dem folgenden Format: `https://{tenantID}.api.crm.dynamics.com`
- `NUM_PARTITIONS`(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes. Beispiel: `2024-01-30T06:47:51.000Z`.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes. Beispiel: `2024-06-30T06:47:51.000Z`.

Einschränkungen

Die folgenden Einschränkungen gelten für den Microsoft Dynamics 365 CRM-Connector:

- In Microsoft Dynamics 365 CRM wird die datensatzbasierte Partitionierung nicht unterstützt, da sie keinen Offset-Parameter unterstützt, sodass die datensatzbasierte Partitionierung nicht unterstützt werden kann.

- Die Paginierung ist auf maximal 500 Datensätze pro Seite festgelegt, um interne Serverausnahmen von SaaS aufgrund einer Kombination aus Datengrößen- und Ratenbeschränkungen zu vermeiden.
  - SaaS-Dokumentation zur Paginierung
  - SaaS-Dokumentation zu Ratenlimits
- Microsoft Dynamics 365 CRM unterstützt nur `order by` übergeordnete Felder für alle Entitäten. `order by`wird für Unterfelder nicht unterstützt.
  - Sowohl ASC- als auch DESC-Richtungen werden unterstützt.
  - `order by`für mehrere Felder wird unterstützt.
- Das Filtern nach dem Feld „createddatetime“ der `aadusers` Standardentität führt zu einem fehlerhaften Anforderungsfehler von SaaS, obwohl es die Filterung unterstützt. Aufgrund des dynamischen Charakters der Metadaten gibt es keine spezifische Identifizierung einer anderen Entität, die ein ähnliches Problem hat, und auch die Ursache ist nicht bekannt. Daher kann es nicht behandelt werden.
- Komplexe Objekttypen wie Struct, List und Map unterstützen keine Filterung.
- Viele Felder, die aus einer Antwort abgerufen werden können, wurden als `false` dynamische Metadatenantwort `isRetrievable` gekennzeichnet. Um Datenverlust zu vermeiden, `isRetrievable` ist `true` für alle Felder auf gesetzt.
- Die feldbasierte Partitionierung wird für alle Entitäten unterstützt, wenn sie die folgenden Kriterien erfüllt:
  - DateTime Abfragbare Felder sollten in Standardentitäten oder `createdon` `modifiedon` Felder (vom System generiert) in benutzerdefinierten Entitäten vorhanden sein.
  - Es gibt keine ausschließliche Identifizierung von systemgenerierten Feldern oder der NULL-Eigenschaft aus SaaS-Metadaten. Es ist jedoch eine allgemeine Praxis APIs, dass nur die standardmäßig verfügbaren Felder filterbar und nicht nullwertbar sind. Daher gilt das obige Kriterium der Feldauswahl als nullsicher, und wenn es filterbar ist, kann es partitioniert werden.

Verbindung zu Microsoft Teams herstellen

Microsoft Teams ist ein kollaborativer Arbeitsbereich innerhalb von Microsoft 365, der als zentraler Knotenpunkt für Gespräche am Arbeitsplatz, kollaborative Teamarbeit, Videochats und den Austausch von Dokumenten dient. All das soll die Produktivität der Mitarbeiter in einer einheitlichen Suite von Tools fördern.

Themen

- AWS Glue Unterstützung für Microsoft Teams
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Microsoft Teams
- Konfiguration von Microsoft Teams-Verbindungen
- Aus Microsoft Teams-Entitäten lesen
- Referenz zu den Verbindungsoptionen für Microsoft Teams
- Einschränkungen
- Erstellen Sie ein neues Microsoft Teams-Konto:

AWS Glue Unterstützung für Microsoft Teams

AWS Glue unterstützt Microsoft Teams wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Microsoft Teams abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Microsoft Teams API-Versionen

v1. Informationen zur Unterstützung von Entitäten pro Version finden Sie unter Unterstützte Entitäten für die Quelle.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Microsoft Teams

Bevor Sie Daten aus Microsoft Teams AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Microsoft Teams-Entwicklerkonto mit E-Mail und Passwort. Weitere Informationen finden Sie unter Erstellen Sie ein neues Microsoft Teams-Konto:.
- Sie sollten in Ihrem Microsoft-Konto eine OAuth2 App eingerichtet haben, die die Client-ID und geheime Anmeldeinformationen bereitstellt, mit denen Sie AWS Glue sicher auf Ihre Daten

zugreifen können, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Erstellen Sie ein neues Microsoft Teams-Konto:.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Microsoft Teams-Konto herstellen. Für typische Verbindungen müssen Sie in Microsoft Teams nichts weiter tun.

Konfiguration von Microsoft Teams-Verbindungen

Microsoft Teams unterstützt die folgenden zwei Typen von Authentifizierungsmechanismen:

1. OAuth Auth: Microsoft Teams unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2
   - Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App verlassen, für die er außer der Microsoft Teams-Instance-URL keine OAuth zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Microsoft Teams weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Microsoft Teams-Instanz gewähren muss.
   - Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Microsoft Teams zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben. In diesem Szenario werden sie weiterhin zu Microsoft Teams weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
   - Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist eine Stunde lang aktiv und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
   - Eine öffentliche Microsoft Teams-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter | Microsoft Learn. Registrieren Sie eine Anwendung bei der Microsoft-Identitätsplattform — Microsoft Graph.

So konfigurieren Sie eine Microsoft Teams-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in einen geheimen Schlüssel zu erstellen AWS Glue.

a. Für die OAuth Authentifizierung:

- Für vom Kunden verwaltete verbundene Apps — Secret sollte das Consumer Secret der verbundenen App mit USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET als Schlüssel enthalten.

2. Stellen Sie unter Datenverbindungen eine AWS Glue Studio Verbindung her, indem Sie die folgenden Schritte ausführen:

a. Wählen Sie unter Datenverbindungen die Option Verbindung erstellen aus.

b. Wählen Sie bei der Auswahl einer Datenquelle Microsoft Teams aus.

c. Geben Sie Ihre Microsoft Teams-Mandanten-ID an.

d. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

e. Stellen Sie die benutzerverwaltete Client-Anwendung ClientId der Microsoft Teams-App bereit.

f. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte. Wählen Sie Weiter.

4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Microsoft Teams-Entitäten lesen

Voraussetzungen

- Ein Microsoft Teams-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Team oder Kanalnachricht. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für Source

Alle Entitäten werden mit API-Version 1.0 unterstützt.

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Teams	Nein	Nein	Nein	Ja	Nein
Mitglieder des Teams	Ja	Ja	Nein	Ja	Ja
Gruppen	Ja	Ja	Ja	Ja	Ja
Mitglieder der Gruppe	Ja	Ja	Nein	Ja	Nein
Kanäle	Ja	Nein	Nein	Ja	Ja
Kanalnachrichten	Nein	Ja	Nein	Ja	Nein
Antworten auf Kanalnachrichten	Nein	Ja	Nein	Ja	Nein
Kanal-Tabs	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Chats	Ja	Ja	Ja	Ja	Ja
Ereignisse im Kalender	Ja	Ja	Ja	Ja	Ja

Beispiel

```
MicrosoftTeams_read = glueContext.create_dynamic_frame.from_options(
    connection_type="MicrosoftTeams",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "v1.0"
    }
```

Microsoft Teams-Entitäts- und Felddetails

Liste der Entitäten:

- Team: https://docs.microsoft.com/en-us/graph/api/user-listet die Teams auf, denen wir beigetreten sind? view=graph-rest-1.0
- Teammitglied: -liste-Mitglieder? https://docs.microsoft.com/en-us/ graph/api/team view=graph-rest-1.0
- Gruppe: -list? https://docs.microsoft.com/en-us/ graph/api/group view=graph-rest-1.0
- Gruppenmitglied: -list-mitglieder? https://docs.microsoft.com/en-us/ graph/api/group view=graph-rest-1.0
- Kanal: -list? https://docs.microsoft.com/en-us/ graph/api/channel view=graph-rest-1.0
- Kanalnachricht: -Meldungen auflisten? https://docs.microsoft.com/en-us/ graph/api/channel view=graph-rest-1.0
- Kanalnachrichten-Antwort: -Antworten auflisten? https://docs.microsoft.com/en-us/ graph/api/chatmessage view=graph-rest-1.0
- Kanal-Tab: -Tabs auflisten? https://docs.microsoft.com/en-us/ graph/api/channel view=graph-rest-1.0

- Chat: -Liste? https://docs.microsoft.com/en-us/ graph/api/chat view=graph-rest-1.0
- Kalenderereignis: Ereignisse auflisten? https://docs.microsoft.com/en-us/ graph/api/group view=graph-rest-1.0

Partitionierung von Abfragen

Zusätzliche Spark-OptionenPARTITION_FIELD,LOWER_BOUND,UPPER_BOUND, NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:"2024-02-06".
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt:

Entity Name	Partitionierung von Feldern	Datentyp
Mitglieder des Teams	visibleHistoryStartDateTime	DateTime
Gruppen	createdDateTime	DateTime
Kanäle	createdDateTime	DateTime
Chats	createdDateTime, lastModif iedDate Zeit	DateTime
Ereignisse im Kalender	createdDateTime, lastModif iedDate Uhrzeit, ursprüngl icher Start	DateTime

Beispiel

```
microsoftteams_read = glueContext.create_dynamic_frame.from_options(
    connection_type="MicrosoftTeams",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "group",
        "API_VERSION": "v1.0",
        "PARTITION_FIELD": "createdDateTime"
        "LOWER_BOUND": "2022-07-13T07:55:27.065Z"
        "UPPER_BOUND": "2022-08-12T07:55:27.065Z"
        "NUM_PARTITIONS": "2"
    }
```

Referenz zu den Verbindungsoptionen für Microsoft Teams

Im Folgenden sind Verbindungsoptionen für Microsoft Teams aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Microsoft Teams.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Microsoft Teams Rest API-Version, die Sie verwenden möchten. Beispiel: v1.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Microsoft Teams-Connector:

- Die Microsoft Teams-API gibt weniger Datensätze zurück als für Chat- und Teammitglieder-Entitäten angegeben. Dieses Problem wurde dem Microsoft Teams-Support gemeldet und wird derzeit untersucht.

Erstellen Sie ein neues Microsoft Teams-Konto:

1. Navigieren Sie zur Startseite von Microsoft Teams https://account.microsoft.com/account/und wählen Sie Anmelden aus.
2. Wählen Sie Create one! .
3. Geben Sie die erforderlichen Informationen für die Kontoerstellung ein und erstellen Sie ein neues Konto.
4. Rufen Sie die Microsoft Teams-Website unter https://www.microsoft.com/en-in/microsoft-teams/log-in auf.
5. Melden Sie sich mit dem Microsoft-Konto an, das Sie gerade erstellt haben.
6. Navigieren Sie nach erfolgreicher Registrierung bei Teams zu https://account.microsft.com/services.
7. Wählen Sie Microsoft 365 testen.
8. Aktivieren Sie eines der folgenden Microsoft 365- oder Microsoft Teams-Abonnements, um auf alle erforderlichen Funktionen von Microsoft Teams Connector zuzugreifen:
   - Grundlagen von Microsoft Teams
   - Microsoft 365 für Unternehmen
   - Microsoft 365 Business Basic
   - Microsoft 365 Standard für Unternehmen
   - Microsoft 365 Business Premium

Erstellen Sie eine verwaltete Client-App:

1. Um eine verwaltete Anwendung zu erstellen, müssen Sie eine neue OAuth App auf Microsoft Entra (früher Azure Active Directory) registrieren:
2. Melden Sie sich im Microsoft Entra Admin Center an.

3. Wenn Sie Zugriff auf mehrere Mandanten haben, verwenden Sie das Einstellungssymbol im oberen Menü, um über das Menü Verzeichnisse + Abonnements zu dem Mandanten zu wechseln, für den Sie die Anwendung registrieren möchten.
4. Navigieren Sie zu Identität > Anwendungen > App-Registrierungen und wählen Sie Neue Registrierung aus.
5. Geben Sie einen Anzeigenamen für Ihre Anwendung ein.
6. Geben Sie im Abschnitt Unterstützte Kontotypen an, wer die Anwendung verwenden kann. Um diese App global zu machen, wählen Sie „Konten in einem beliebigen Organisationsverzeichnis“ oder „Konten in einem beliebigen Organisationsverzeichnis und persönliche Microsoft-Konten“.
7. Geben Sie die Weiterleitungs-URL ein`https://{region}.console.aws.amazon.com/appflow/oauth`. Fügen Sie beispielsweise für den `us-west-2 region` hinzu`https://us-west-2.console.aws.amazon.com/appflow/oauth`. Sie können mehrere URLs für verschiedene Regionen hinzufügen, die Sie verwenden möchten.
8. Registrieren Sie die App.
9. Notieren Sie sich die Client-ID für die future Verwendung.
10. Wählen Sie im Abschnitt Essentials die Option Zertifikat oder Geheimnis hinzufügen aus.
11. Wählen Sie New Client Secret aus.
12. Geben Sie Beschreibung und Gültigkeitsdauer ein.
13. Kopieren Sie das Client-Geheimnis und speichern Sie es für die future Verwendung.
14. Wählen Sie in der Menüliste auf der linken Seite API-Berechtigungen aus.
15. Wählen Sie „Eine Berechtigung hinzufügen“.
16. Wählen Sie „Microsoft Graph“.
17. Wählen Sie „Delegierte Berechtigungen“.
18. Überprüfen Sie alle folgenden Berechtigungen:

    - Benutzer.Lesen
    - Offline_Zugriff
    - Benutzer.Lesen.Alle
    - Benutzer. ReadWrite. Alle
    - TeamsTab.ReadWriteForTeam
    - TeamsTab.ReadWriteForChat
    - TeamsTab. ReadWrite. Alle

- TeamsTab.Lesen.Alle
- TeamSettings. ReadWrite. Alle
- TeamSettings.Lesen.Alle
- TeamMember. ReadWrite. Alle
- TeamMember.Lesen.Alle
- Mannschaft. ReadBasic. Alle
- GroupMember. ReadWrite. Alle
- GroupMember.Lesen.Alle
- Gruppe. ReadWrite. Alle
- Gruppe.Lesen.Alle
- Verzeichnis. ReadWrite. Alle
- Verzeichnis.Lesen.Alles
- Verzeichnis. AccessAsUser. Alle
- Chatten. ReadWrite
- Chatten. ReadBasic
- Chatten. Lesen
- ChannelSettings. ReadWrite. Alle
- ChannelSettings.Lesen.Alle
- ChannelMessage.Lesen.Alles
- Kanal. ReadBasic. Alle

19. Wählen Sie Add permissions (Berechtigungen hinzufügen) aus. Ihre App ist jetzt erfolgreich eingerichtet. Sie können die Client-ID und das Client-Geheimnis verwenden, um eine neue Verbindung herzustellen. Weitere Informationen finden Sie in https://learn.microsoft.com/en-us/graph/ auth-register-app-v 2.

Verbindung zu Mixpanel herstellen

Mixpanel ist eine leistungsstarke Echtzeit-Analyseplattform, mit der Unternehmen das Nutzerengagement messen und optimieren können. Mixpanel ist eine App zur Verfolgung des Kundenverhaltens. Sie können damit verfolgen, wie Benutzer mit Ihrem Produkt interagieren, und diese Daten mit interaktiven Berichten analysieren, mit denen Sie die Ergebnisse mit nur wenigen Klicks abfragen und visualisieren können. Als Mixpanel-Benutzer können Sie sich mit Ihrem

Mixpanel-Konto verbinden AWS Glue . Anschließend können Sie Mixpanel als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Mixpanel und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Mixpanel
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Mixpanel konfigurieren
- Mixpanel-Verbindungen konfigurieren
- Aus Mixpanel-Entitäten lesen
- Mixpanel-Verbindungsoptionen
- Ein Mixpanel-Konto erstellen und die Client-App konfigurieren
- Einschränkungen

AWS Glue Unterstützung für Mixpanel

AWS Glue unterstützt Mixpanel wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Mixpanel abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Mixpanel-API-Versionen

2.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Mixpanel konfigurieren

Bevor Sie die Übertragung von Mixpanel verwenden AWS Glue können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein Mixpanel-Konto. Weitere Informationen zum Erstellen eines Kontos finden Sie unter Erstellen eines Mixpanel-Kontos.
- Ihr Mixpanel-Konto ist für den API-Zugriff aktiviert. Der API-Zugriff ist standardmäßig für die Enterprise, Unlimited, Developer und Performance Editionen aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem Mixpanel-Konto verbinden AWS Glue . Für typische Verbindungen müssen Sie in Mixpanel nichts weiter tun.

Mixpanel-Verbindungen konfigurieren

Mixpanel unterstützt Benutzername und Passwort für. `BasicAuth` Die Standardauthentifizierung ist eine einfache Authentifizierungsmethode, bei der Kunden Anmeldeinformationen direkt eingeben, um auf geschützte Ressourcen zuzugreifen. AWS Glue ist in der Lage, den Benutzernamen und das Passwort zur Authentifizierung von APIs Mixpanel zu verwenden.

Die öffentliche Mixpanel-Dokumentation zu `BasicAuth` Flow finden Sie unter Mixpanel-Dienstkonten.

Um eine Mixpanel-Verbindung zu konfigurieren:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Für die Standardauthentifizierung sollte Secret die verbundene App Consumer Secret mit `USERNAME` und `PASSWORD` als Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, pro Verbindung in ein Secret zu erstellen AWS Glue.

2. Stellen Sie im unter Datenverbindungen eine Verbindung her AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Mixpanel aus.
   b. Geben Sie das Mixpanel `INSTANCE_URL` an, zu dem Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
```

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie Netzwerkoptionen, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Mixpanel-Entitäten lesen

Voraussetzungen

Sie benötigen ein Mixpanel-Objekt wie, `FunnelsRetention`, oder`Retention Funnels`, aus dem Sie Daten lesen möchten. Außerdem müssen Sie den Objektnamen kennen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Trichter	Ja	Nein	Nein	Ja	Nein
Aufbewahrung	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Segmentierung	Ja	Nein	Nein	Ja	Nein
Summe der Segmentierung	Ja	Nein	Nein	Ja	Nein
Durchschnitt der Segmentierung	Ja	Nein	Nein	Ja	Nein
Kohorten	Ja	Nein	Nein	Ja	Nein
Engagieren	Nein	Ja	Nein	Ja	Nein
--Ereignisse	Ja	Nein	Nein	Ja	Nein
Top Veranstaltungen	Ja	Nein	Nein	Ja	Nein
Namen der Ereignisse	Ja	Nein	Nein	Ja	Nein
Eigenschaften von Ereignissen	Ja	Nein	Nein	Ja	Nein
Eigenschaften von Veranstaltungen Top	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Eigenschaften und Werte für Ereignisse	Ja	Nein	Nein	Ja	Nein
Anmerkungen	Ja	Nein	Nein	Ja	Nein
Aktivität eines Profilereignisses	Ja	Nein	Nein	Ja	Nein

Beispiel

```
mixpanel_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mixpanel",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/cohorts/list?project_id=2603353",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://www.mixpanel.com/api/app/me"
    }
```

Details zur Mixpanel-Entität und zum Feld

Entität	Feld	Datentyp	Unterstützte Betreiber
Trichter	funnel_id	Ganzzahl	'='
	Arbeitsbereichs-ID	Ganzzahl	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'=’

Entität	Feld	Datentyp	Unterstützte Betreiber
	length	Ganzzahl	'='
	Länge_Einheit	String	'='
	Intervall	Ganzzahl	'='
	Einheit	String	'='
	limit	Ganzzahl	'='
	data	Struct	
	Meta	Struct	
Aufbewahrung	Arbeitsbereichs-ID	Ganzzahl	'='
	Einheit	String	'='
	Einheit für Sucht	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	event	String	'='
	limit	Ganzzahl	'='
	data	Struct	
Segmentierung	Arbeitsbereichs-ID	Ganzzahl	'='
	event	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	Einheit	String	'='

Entität	Feld	Datentyp	Unterstützte Betreiber
	Intervall	Ganzzahl	'='
	limit	Ganzzahl	'='
	Typ	String	'='
	Serie	Auflisten	
	values	Struct	
	data	Struct	
Segmentierung Numerisch	workspace_id	Ganzzahl	'='
	event	String	'='
	on	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	Einheit	String	'='
	Typ	String	'='
	Serie	Auflisten	
	values	Struct	
Summe der Segmentierung	Arbeitsbereichs-ID	Ganzzahl	'='
	event	String	'='
	on	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='

Entität	Feld	Datentyp	Unterstützte Betreiber
	Einheit	String	'='
	Metadaten	Struct	
	results	Struct	
Durchschnitt der Segmentierung	workspace_id	Ganzzahl	'='
	event	String	'='
	on	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	Einheit	String	'='
	Metadaten	Struct	
	results	Struct	
Kohorten	count	Ganzzahl	
	ist_sichtbar	Ganzzahl	
	description	String	
	created	DateTime	
	projekt-ID	Ganzzahl	
	id	BigInteger	
	Name	String	
	Datengruppen-ID	String	
Engagieren	distinct_id	String	

Entität	Feld	Datentyp	Unterstützte Betreiber
		Eigenschaften	Struct
Ereignis	Workspace	Ganzzahl	'='
	event	String	'='
	Typ	String	'='
	Einheit	String	'='
	Intervall	Ganzzahl	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	Serie	Auflisten	
	values	Struct	
Top Veranstaltungen	Typ	String	'='
	workspace_id	Ganzzahl	'='
	limit	Ganzzahl	'='
	Betrag	Ganzzahl	
	event	String	
	Prozentuale Änderung	Gleitkommazahl	
Ereignisname	data	Auflisten	
	Arbeitsbereichs-ID	Ganzzahl	'='
	Typ	String	'='
	limit	Ganzzahl	'='

Entität	Feld	Datentyp	Unterstützte Betreiber
Ereigniseigenschaften	Arbeitsbereichs-ID	Ganzzahl	'='
	event	String	'='
	Name	String	'='
	Typ	String	'='
	Einheit	String	'='
	Intervall	Ganzzahl	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	limit	Ganzzahl	'='
	data	Struct	
	Serie	Auflisten	
	values	Struct	
Eigenschaften der Veranstaltung: Top	workspace_id	Ganzzahl	'='
	event	String	'='
	limit	Ganzzahl	'='
	data	Struct	
Wert der Eigenschaften des Ereignisses	workspace_id	Ganzzahl	'='
	event	String	'='
	limit	Ganzzahl	'='
	Name	String	'='

Entität	Feld	Datentyp	Unterstützte Betreiber
	data	Auflisten	
Anmerkung	Arbeitsbereichs-ID	Ganzzahl	
	date	DateTime	
	projekt-ID	Ganzzahl	
	id	BigInteger	
	description	String	
	ab_Datum	Datum	BETWEEN
Profil, Ereignis, Aktivität	workspace_id	Ganzzahl	'='
	eindeutige_IDs	String	'='
	ab_Datum	Datum	'='
	bis_datum	Datum	'='
	event	String	
	Eigenschaften	Struct	

Mixpanel-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Mixpanel aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Mixpanel.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Mixpanel Rest API-Version, die Sie verwenden möchten. Zum Beispiel: v2.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.

- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Ein Mixpanel-Konto erstellen und die Client-App konfigurieren

Ein Mixpanel-Konto erstellen

1. Navigieren Sie zur Mixpanel-Homepage/.
2. Wählen Sie auf der Mixpanel-Startseite in der oberen rechten Ecke der Seite die Option Sign Up aus.
3. Führe auf der Seite „Los geht's los“ die folgenden Aktionen aus:

   - Geben Sie Ihre E-Mail-Adresse in das dafür vorgesehene Feld ein.
   - Wählen Sie das erforderliche Kontrollkästchen aus, um den Bedingungen zuzustimmen.
   - Wählen Sie Erste Schritte, um fortzufahren.

   Nach erfolgreichem Abschluss erhalten Sie eine Bestätigungs-E-Mail.
4. Suchen Sie in Ihrem E-Mail-Posteingang nach einer Bestätigungsnachricht, öffnen Sie die E-Mail und folgen Sie den Anweisungen zur Bestätigung Ihrer E-Mail-Adresse.
5. Wählen Sie auf der Bestätigungsseite die Option E-Mail verifizieren aus, um Ihre E-Mail-Bestätigung abzuschließen.
6. Geben Sie auf der Seite Name Your Organization den Namen Ihrer Organisation ein und klicken Sie auf Weiter.
7. Geben Sie auf der Seite Ihr erstes Projekt Ihre Projektdetails ein und wählen Sie Erstellen aus.
8. Wählen Sie auf der nächsten Seite Let's get Started aus, um die Erstellung Ihres Kontos abzuschließen.

Loggen Sie sich in ein Mixpanel-Konto ein

1. Navigieren Sie zur Mixpanel-Anmeldeseite/.
2. Geben Sie Ihre E-Mail-Adresse ein und wählen Sie Weiter.
3. Suchen Sie in Ihrem E-Mail-Posteingang nach einer Bestätigungsnachricht, öffnen Sie die E-Mail und folgen Sie den Anweisungen, um Ihre E-Mail-Adresse zu verifizieren.

4. Wählen Sie auf der nächsten Seite die Schaltfläche Anmelden, um sich bei Ihrem Konto anzumelden.

Kauf eines Mixpanel-Plans

1. Wählen Sie auf der Mixpanel-Seite das Einstellungssymbol in der oberen rechten Ecke der Seite aus.
2. Wählen Sie in der Liste der Optionen die Option Abo-Details und Abrechnung aus.
3. Wählen Sie auf der Seite Abo-Details und Abrechnung die Option Upgrade oder Ändern aus.
4. Wählen Sie auf der nächsten Seite den Plan aus, den Sie kaufen möchten.

   Damit sind die Kontoerstellung und der Kauf des Plans abgeschlossen.

Erstellung eines Benutzernamens und eines geheimen Kundengeheimnisses (um deine App zu registrieren)

1. Wählen Sie auf der Mixpanel-Seite das Einstellungssymbol in der oberen rechten Ecke der Seite aus.“
2. Wählen Sie aus der Liste der Optionen die Option Projekteinstellungen aus.
3. Wählen Sie auf der Seite mit den Projekteinstellungen die Option Dienstkonten und dann Dienstkonto hinzufügen aus.
4. Wählen Sie in der Dropdownliste Dienstkonto das Dienstkonto aus oder geben Sie den Namen ein, den Sie erstellen möchten, fügen Sie die Projektrolle hinzu, geben Sie die Gültigkeitsdauer an und wählen Sie Hinzufügen aus.

   > **Important**
   >
   > Nach Abschluss des vorherigen Schritts wird auf der folgenden Seite der geheime Schlüssel des Dienstkontos angezeigt. Stellen Sie sicher, dass Sie den geheimen Schlüssel des Dienstkontos speichern. Nach diesem Zeitpunkt können Sie nicht mehr darauf zugreifen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Mixpanel-Anschluss:

- Bei `Segmentation Numeric` Entität gibt die Mixpanel-API einen `400 (Bad Request)` Fehler aus, wenn keine numerischen Daten für die obligatorischen Filter gefunden werden. Wir behandeln dies als `0K` Reaktion, um Flussausfälle zu verhindern.
- Das abfragbare Feld `limit` wurde aus folgenden Gründen aus den unterstützten Entitäten entfernt:
  - Es verursachte Fehler, weil es als Limit-Funktion des SDK interpretiert wurde
  - Der Filter erfüllte keinen praktischen Zweck
  - Entsprechende Funktionen werden nun durch die Implementierung der Limit-Feature abgedeckt
- Feldbasierte Partitionierung kann nicht unterstützt werden, da keine erforderlichen Operatoren (>=,,, <= <>,`between`) für die Partitionierung von der SaaS-Plattform aus erforderlich sind. Sie unterstützt zwar den `between` Operator, aber die Felder, für die sie diesen Operator unterstützt, sind nicht abrufbar. Daher sind die Kriterien für die feldbasierte Partitionierung nicht erfüllt.
- Da es nicht vorgesehen ist, einen Offset-Wert für Entitäten zu erhalten, die Paginierung unterstützen, ist es nicht möglich, die datensatzbasierte Partitionierung für Mixpanel zu unterstützen.
- `Cohorts`Die Entität unterstützt nur `CreatedDate/Time` Felder und es gibt kein Feld, das identifiziert werden kann, `UpdatedDate/Time` da das Ergebnis `DML_Status` nicht identifiziert werden kann. Außerdem gibt es keinen Endpunkt zur Identifizierung gelöschter Datensätze. Daher kann CDC nicht unterstützt werden.
- Um einen AWS Glue Job für die unten genannten Entitäten auszuführen, sind obligatorische Filter erforderlich. In der folgenden Tabelle finden Sie die Namen der Entitäten und die erforderlichen Filter.

  Entitätsname und erforderliche Filter

Entity-Name	Obligatorische Filter
Anmerkungen	from_date, to_date
Kohorten	Keine
Engagieren	Keine
Ereignis	Ereignis, Typ, Einheit, from_date, to_date, am
Name des Ereignisses	Typ

Entity-Name	Obligatorische Filter
Eigenschaften der Ereignisse	Ereignis, Name, Typ, Einheit, from_date, to_date
Eigenschaften von Ereignissen Am Anfang	event
Eigenschaften und Werte für Ereignisse	Ereignis, Name
Veranstaltungen Top	Typ
Trichter	trichter_id, from_date, to_date
Profil, Ereignis, Aktivität	distinct_ids, from_date, to_date
Aufbewahrung	from_date, to_date, unit, addiction_unit
Segmentierung	Ereignis, From_Date, To_Date
Durchschnitt der Segmentierung	Ereignis, from_date, to_date, am
Segmentierung Numerisch	Ereignis, from_date, to_date, am
Summe der Segmentierung	Ereignis, from_date, to_date, am

Verbindung zum Montag herstellen

Monday.com ist ein vielseitiges Arbeitsbetriebssystem, das das Projektmanagement und die Teamzusammenarbeit optimiert. Es bietet anpassbare Workflows, visuelle Dashboards und Automatisierungstools zur Steigerung der Produktivität. Benutzer können Aufgaben verfolgen, Ressourcen verwalten und effektiv auf einer integrierten Plattform kommunizieren.

Themen

- AWS Glue Support für Montag
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Montag konfigurieren
- Monday-Verbindungen konfigurieren
- Aus Monday-Entitäten lesen

- Referenz zur Verbindungsoption am Montag
- Einschränkungen
- Erstellen Sie ein neues Monday-Konto:

AWS Glue Support für Montag

AWS Glue unterstützt Montag wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten ab Montag abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Monday-API-Versionen

v2.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Montag konfigurieren

Bevor Sie Daten ab Montag AWS Glue übertragen können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein Monday-Entwicklerkonto mit E-Mail und Passwort. Weitere Informationen finden Sie unter Erstellen Sie ein neues Monday-Konto:.
- Ihr Monday-Entwicklerkonto ist für den API-Zugriff aktiviert. Die gesamte Nutzung von The Monday APIs ist innerhalb der Testphase ohne zusätzliche Kosten möglich. Nach Ablauf der Testphase müssen Sie ein Abonnement erwerben, um Daten zu erstellen und darauf zuzugreifen. Weitere Informationen finden Sie auf der Lizenzierungsseite vom Montag.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Monday-Konto herstellen. Für typische Verbindungen müssen Sie am Montag nichts weiter tun.

Monday-Verbindungen konfigurieren

Monday unterstützt die folgenden zwei Typen von Authentifizierungsmechanismen:

1. OAuth Auth: Monday unterstützt den Gewährungstyp AUTHORIZATION_CODE für. OAuth2
   - Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue verbundene App verlassen, für die er außer der Instanceurl am Montag keine OAuth weiteren Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Monday weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Monday-Instanz gewähren muss.
   - Benutzer sollten sich dafür entscheiden, am Montag ihre eigene verbundene App zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Monday weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
   - Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist eine Stunde lang aktiv und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
   - Weitere Informationen finden Sie in der Dokumentation zum Erstellen einer verbundenen App für den AUTHORIZATION_CODE-Flow OAuth .
2. Benutzerdefinierte Authentifizierung:
   - Eine öffentliche Montagsdokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie unter https://developer.monday.com/api-reference/api-token-permissionsdocs/authentication#.

So konfigurieren Sie eine Monday-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in einen geheimen Schlüssel zu erstellen AWS Glue.
   a. Für die OAuth Authentifizierung:
      - Für vom Kunden verwaltete verbundene Apps — Secret sollte das Consumer Secret der verbundenen App mit USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET als Schlüssel enthalten.
   b. Für benutzerdefinierte Authentifizierung:

- Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `personalAccessToken` Schlüssel enthalten.

2. Stellen Sie in AWS Glue Studio unter Datenverbindungen eine Verbindung her, indem Sie die folgenden Schritte ausführen:

   a. Wählen Sie unter Datenverbindungen die Option Verbindung erstellen aus.

   b. Wählen Sie bei der Auswahl einer Datenquelle Montag aus.

   c. Geben Sie Ihre Instance-URL für Montag ein.

   d. Wählen Sie die IAM-Rolle aus, die die folgenden AWS Glue Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   e. Wählen Sie den Authentifizierungstyp, um eine Verbindung zu Monday herzustellen

   - Für OAuth die Authentifizierung: Geben Sie die Token-URL und die benutzerverwaltete Client-Anwendung für den Montag ClientId an, zu dem Sie eine Verbindung herstellen möchten.
   - Für benutzerdefinierte Authentifizierung: Wählen Sie den Authentifizierungstyp CUSTOM aus, um eine Verbindung zu Monday herzustellen.

   f. Wählen Sie den aus`secretName`, den Sie für diese Verbindung verwenden möchten AWS Glue , um die Token zu platzieren.

   g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte. Wählen Sie Weiter.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Monday-Entitäten lesen

Voraussetzungen

- Ein Monday-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten für Source

Liste der Entitäten:

- Konto: https://developer.monday.com/api-reference/docs/account #queries
- Forum: docs/boards #queries https://developer.monday.com/api-reference/
- Spalte: Dokumente/Spalten #queries https://developer.monday.com/api-reference/
- Dokumente: Dokumente/Dokumente #queries https://developer.monday.com/api-reference/
- Dokumentenblock: https://developer.monday.com/api-reference/ Dokumente/Blöcke #queries
- Dateien: Dokumente/Dateien #queries https://developer.monday.com/api-reference/
- Ordner: Dokumente/Ordner #queries https://developer.monday.com/api-reference/
- Gruppen: Dokumente/Gruppen #queries https://developer.monday.com/api-reference/
- Artikel: Dokumente/Artikel #queries https://developer.monday.com/api-reference/
- Unterelemente: Dokumente/Unterelemente #queries https://developer.monday.com/api-reference/
- Schlagworte: docs/tags-queries #queries https://developer.monday.com/api-reference/
- Teams: Dokumente/Teams #queries https://developer.monday.com/api-reference/
- Aktualisierungen: Dokumente/Updates #queries https://developer.monday.com/api-reference/
- Benutzer: Dokumente/Benutzer #queries https://developer.monday.com/api-reference/
- Arbeitsbereiche: Dokumente/Arbeitsbereiche #queries https://developer.monday.com/api-reference/

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Account	Nein	Nein	Nein	Ja	Nein
Bretter	Ja	Ja	Nein	Ja	Nein
Spalten	Nein	Nein	Nein	Ja	Nein
Dokumente	Ja	Ja	Nein	Ja	Nein
Dokumentenblöcke	Nein	Ja	Nein	Ja	Nein
Dateien	Ja	Nein	Nein	Ja	Nein
Gruppen	Nein	Nein	Nein	Ja	Nein
Item	Ja	Ja	Nein	Ja	Nein
Unterelemente	Nein	Nein	Nein	Ja	Nein
Tags	Ja	Nein	Nein	Ja	Ja
Teams	Ja	Nein	Nein	Ja	Nein
Aktualisierungen	Nein	Ja	Nein	Ja	Nein
Benutzer	Ja	Ja	Nein	Ja	Nein
Workspaces	Ja	Ja	Nein	Ja	Nein
Ordner	Ja	Ja	Nein	Ja	Nein

Beispiel

```
monday_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="monday",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "account",
        "API_VERSION": "v2"
    }
```

Referenz zur Verbindungsoption am Montag

Im Folgenden sind die Verbindungsoptionen für Montag aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts am Montag.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Monday Rest API-Version, die Sie verwenden möchten. Beispiel: v2.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Einschränkungen

Die folgenden Einschränkungen gelten für den Monday-Connector:

- Die dynamische Metadatenantwort weist einige Konflikte mit der Dokumentation auf, wie unten beschrieben:
  - Die Entität Group, Column unterstützt Filteroperationen, ist jedoch nicht im Endpunkt dynamischer Metadaten vorhanden und wird daher als nicht filterbare Entität beibehalten.
  - Der dynamische Endpunkt besteht aus mehr als 15.000 Zeilen und gibt Metadaten aller Entitäten in einer einzigen Antwort zurück. Aus diesem Grund dauert das Laden der Felder durchschnittlich 10 Sekunden, sodass dies bei der Ausführung eines Jobs etwas mehr Zeit in Anspruch nehmen würde.
  - In der folgenden Tabelle finden Sie das Ratenlimit für Montag. Die beträchtliche Größe der Antwortdaten der dynamischen Entität führt zu einer spürbaren Verzögerung. Das Laden von Feldern dauert durchschnittlich 10 Sekunden.

Limitierung der Komplexität	5.000.000 (5 Millionen) Komplexitätspunkte
Tägliches Anruflimit	10.000 für den Pro-Plan
Minutenlimit	500 Abfragen pro Minute
Limit für Parallelität	Maximal 100 gleichzeitige Anfragen für Pro Plan

Erstellen Sie ein neues Monday-Konto:

1. Navigiere zur Startseite von Monday, https://monday.com/und wähle Anmelden.
2. Sie werden zur Anmeldeseite weitergeleitet. Wählen Sie unten auf der Seite die Option Anmelden aus.
3. Geben Sie Ihre E-Mail-Adresse ein und wählen Sie Weiter. Alternativ können Sie sich bei Google anmelden.
4. Geben Sie die erforderlichen Daten ein und wählen Sie Weiter.
5. Füllen Sie die Fragen der Umfrage aus und folgen Sie den Schritten, um die Kontoerstellung abzuschließen.

Registrieren Sie eine OAuth Anwendung:

1. Logge dich in dein monday.com-Konto ein. Klicke auf deinen Avatar (Bildsymbol) in der unteren linken Ecke deines Bildschirms.
2. Wählen Sie Entwickler.
3. Wählen Sie App erstellen.
4. Füllen Sie die erforderlichen Felder für Name und Beschreibung aus.
5. Navigieren Sie zum Abschnitt „OAuth“ auf der rechten Seite, fügen Sie die Bereiche hinzu und wählen Sie „Funktion speichern“.
6. Navigieren Sie zum Tab „URLS umleiten“ neben dem Bereich, fügen Sie die Weiterleitungs-URL hinzu und wählen Sie „Funktion speichern“.
7. Geben Sie auf der URLs Registerkarte „Weiterleitung“ die URL Ihrer App ein. Das sollte https://{region-code}.console.aws.amazon.com/appflow/ OAuth sein. Wenn Sie

beispielsweise verwenden, können `us-east-1` Sie hinzufügen. `https://us-east-1.console.aws.amazon.com/appflow/oauth`

8. Die Anwendung ist jetzt einsatzbereit. Sie finden Ihre Anmeldeinformationen im Abschnitt „Grundinformationen“. Notieren Sie sich Ihre Client-ID und Ihre geheimen Client-Zeichenketten. Diese Zeichenfolgen werden verwendet, um über einen AppFlow Connector eine Verbindung mit dieser App herzustellen.

Generieren Sie ein persönliches Zugriffstoken:

Derzeit bietet monday.com nur unsere V2-API-Token an, bei denen es sich ausschließlich um persönliche Token handelt. Um auf deine API-Token zuzugreifen, kannst du je nach Benutzerebene eine von zwei Methoden verwenden. Admin-Benutzer können beide Methoden verwenden, um ihre API-Token zu erwerben. Mitgliedsbenutzer können über ihre Entwickler-Tabs auf ihre API-Token zugreifen.

Admins — Wenn du ein Admin-Nutzer deines monday.com-Kontos bist, kannst du über den Tab „Admin“ mit den folgenden Schritten auf deine API-Token zugreifen:

1. Logge dich in dein monday.com-Konto ein. Klicke auf deinen Avatar (Bildsymbol) in der unteren linken Ecke deines Bildschirms.
2. Wählen Sie im daraufhin angezeigten Menü „Administration“ aus (dafür benötigen Sie Administratorrechte).
3. Navigieren Sie zum Abschnitt „API“ und generieren Sie ein „API V2-Token“. Sie können Ihr Token kopieren und verwenden.

Entwickler — Wenn du ein Mitglied deines monday.com-Kontos bist, kannst du über den Entwickler-Tab mit den folgenden Schritten auf deine API-Token zugreifen:

1. Logge dich in dein monday.com-Konto ein. Klicke auf deinen Avatar (Bildsymbol) in der unteren linken Ecke deines Bildschirms.
2. Wählen Sie im daraufhin angezeigten Menü „Entwickler“ aus.
3. Wählen Sie im oberen Menü das Drop-down-Menü „Entwickler“. Wählen Sie im Drop-down-Menü die erste Option mit dem Titel „Meine Zugriffstoken“ aus.

Verbindung zu MongoDB herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für MongoDB. AWS Glue Studio bietet eine visuelle Schnittstelle, um eine Verbindung zu MongoDB herzustellen, Datenintegrationsjobs zu erstellen und sie auf dem AWS Glue Studio serverlose Spark-Laufzeit.

Themen

- Erstellen einer MongoDB-Verbindung
- Erstellen eines MongoDB-Quellknotens
- Erstellen eines MongoDB-Zielknotens
- Erweiterte Optionen

Erstellen einer MongoDB-Verbindung

Voraussetzungen:

- Wenn sich Ihre MongoDB-Instance in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue Job mit der MongoDB-Instance kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die bei der Ausführung des Jobs verwendet AWS Glue werden. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer MongoDB-Instance und diesem Standort zugelassen wird. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

Eine Verbindung zu MongoDB konfigurieren Sie wie folgt:

1. Erstellen Sie optional ein Geheimnis mit Ihren MongoDB-Anmeldeinformationen. AWS Secrets Manager Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `username` mit dem Wert. *`mongodbUser`*

Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *`mongodbPass`*

2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen,*`connectionName`*, für die future Verwendung in AWS Glue.

   - Wählen Sie als Verbindungstyp die Option MongoDB oder MongoDB Atlas aus.
   - Geben Sie als MongoDB-URL oder MongoDB-Atlas-URL den Hostnamen Ihrer MongoDB-Instance an.

     Eine MongoDB-URL wird im Format `mongodb://`*`mongoHost`*`:`*`mongoPort`*`/`*`mongoDBname`* bereitgestellt.

     Eine MongoDB-Atlas-URL wird im Format `mongodb +srv://`*`mongoHost`*`:`*`mongoPort`*`/`*`mongoDBname`* bereitgestellt.

     Die Bereitstellung der Standarddatenbank für die Verbindung *`mongoDBname`* ist optional.
   - Wenn Sie sich dafür entschieden haben, ein Secrets Manager Manager-Geheimnis zu erstellen, wählen Sie den Typ AWS Secrets Manager Credential.

     Geben Sie dann im Feld AWS Secret Folgendes ein*`secretName`*.
   - Wenn Sie sich dafür entscheiden, einen Benutzernamen und ein Passwort anzugeben, geben Sie *`mongodbUser`* und ein*`mongodbPass`*.
3. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für MongoDB-Instances, die AWS in einer Amazon VPC gehostet werden
     - Sie müssen Amazon VPC-Verbindungsinformationen für die Verbindung angeben, die Ihre AWS Glue MongoDB-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Nachdem Sie eine AWS Glue MongoDB-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue Job ausführen:

- Wenn Sie mit AWS Glue Jobs im Visual Editor arbeiten, müssen Sie Amazon VPC-Verbindungsinformationen angeben, damit Ihr Job eine Verbindung zu MongoDB herstellen kann.

Identifizieren Sie einen geeigneten Standort in Amazon VPC und geben Sie ihn in Ihrer AWS Glue - MongoDB-Verbindung an.

- Wenn Sie sich dafür entschieden haben, ein Secrets Manager Manager-Geheimnis zu erstellen, gewähren Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle *secretName* Leserechte.

Erstellen eines MongoDB-Quellknotens

Voraussetzungen

- Eine AWS Glue MongoDB-Verbindung, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer MongoDB-Verbindung”.
- Wenn Sie ein Secrets-Manager-Secret erstellt haben, benötigen Sie die entsprechenden Berechtigungen für Ihren Auftrag, um das von der Verbindung verwendete Geheimnis zu lesen.
- Eine MongoDB-Sammlung, aus der Sie lesen möchten. Sie benötigen Identifikationsinformationen für die Sammlung.

  Eine MongoDB-Sammlung wird durch einen Datenbanknamen und einen Sammlungsnamen identifiziert,*mongodbName*,*mongodbCollection*.

Hinzufügen einer MongoDB-Datenquelle

Fügen Sie einen Knoten Datenquelle – MongoDB wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre MongoDB-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie MongoDB-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer MongoDB-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie eine Datenbank aus. Geben Sie *mongodbName* ein.
3. Wählen Sie eine Sammlung aus. Geben Sie *mongodbCollection* ein.
4. Wählen Sie einen Partitionierer, eine Partitionsgröße (MB) und einen Partitionsschlüssel aus. Weitere Informationen zu Partitionierungsparametern finden Sie unter the section called “„connectionType“: „mongodb“ als Quelle”.

5. Geben Sie unter Benutzerdefinierte MongoDB-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines MongoDB-Zielknotens

Voraussetzungen

- Eine AWS Glue MongoDB-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer MongoDB-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine MongoDB-Tabelle, in die Sie schreiben möchten,*`tableName`*.

Hinzufügen eines MongoDB-Datenziels

Fügen Sie einen Knoten Datenziel – MongoDB wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre MongoDB-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie MongoDB-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer MongoDB-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie eine Datenbank aus. Geben Sie *`mongodbName`* ein.
3. Wählen Sie eine Sammlung aus. Geben Sie *`mongodbCollection`* ein.
4. Wählen Sie einen Partitionierer, eine Partitionsgröße (MB) und einen Partitionsschlüssel aus. Weitere Informationen zu Partitionierungsparametern finden Sie unter the section called “„connectionType“: „mongodb“ als Quelle”.
5. Wählen Sie bei Bedarf Schreibvorgänge wiederholen aus.
6. Geben Sie unter Benutzerdefinierte MongoDB-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines MongoDB-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie die, die bei der Programmierung AWS Glue für Spark-Skripte verfügbar sind.

Siehe the section called “MongoDB-Verbindung”.

Verbindung zu Oracle herstellen NetSuite

Oracle NetSuite ist eine all-in-one Cloud-Business-Management-Lösung, die Unternehmen dabei unterstützt, effektiver zu arbeiten, indem sie Kernprozesse automatisiert und Echtzeiteinblicke in die betriebliche und finanzielle Leistung bietet. Mit einer einzigen, integrierten Suite von Anwendungen für die Verwaltung von Buchhaltung, Auftragsabwicklung, Bestandsverwaltung, Produktion, Lieferkette und Lagerbetrieb NetSuite bietet Oracle Unternehmen einen klaren Überblick über ihre Daten und eine strengere Kontrolle über ihre Geschäfte.

Themen

- AWS Glue Unterstützung für Oracle NetSuite
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Oracle NetSuite
- Konfiguration von NetSuite Oracle-Verbindungen
- Aus NetSuite Oracle-Entitäten lesen
- NetSuite Oracle-Verbindungsoptionen
- Einschränkungen und Hinweise für Oracle NetSuite Connector

AWS Glue Unterstützung für Oracle NetSuite

AWS Glue unterstützt Oracle NetSuite wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Oracle abzufragen NetSuite.

Wird es als Ziel unterstützt?

Nein.

Unterstützte NetSuite Oracle-API-Versionen

Die folgenden NetSuite Oracle-API-Versionen werden unterstützt:

- v1

Informationen zur versionsspezifischen Unterstützung von Entitäten finden Sie unter Unterstützte Entitäten für Source.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Oracle NetSuite

Bevor Sie Daten von Oracle AWS Glue übertragen können NetSuite, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein NetSuite Oracle-Konto. Weitere Informationen finden Sie unter Ein NetSuite Oracle-Konto erstellen.
- Ihr NetSuite Oracle-Konto ist für den API-Zugriff aktiviert.
- Sie haben eine OAuth 2.0-API-Integration in Ihrem NetSuite Oracle-Entwicklerkonto erstellt. Diese Integration stellt die Client-Anmeldeinformationen bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Aufrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Eine NetSuite Oracle-Client-App und OAuth 2.0-Anmeldeinformationen erstellen.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem NetSuite Oracle-Konto herstellen.

Ein NetSuite Oracle-Konto erstellen

Navigieren Sie zu Oracle NetSuite und wählen Sie Free Product Tour. Geben Sie die erforderlichen Details ein, um eine kostenlose Produkttour zu erhalten, über die Sie einen Anbieter kontaktieren können. Das Verfahren zur Beantragung eines Kontos ist wie folgt:

- Die Beschaffung eines NetSuite Kontos erfolgt über einen Anbieter, der ein Formular/Angebot erstellt, das rechtlich geprüft werden muss.
- Das Konto, das für den Oracle NetSuite Connector beschafft werden soll, ist vom Standard Cloud Service.

- Dieses Konto wird vom Anbieter erstellt und temporäre Anmeldeinformationen werden von ihm gemeinsam genutzt. Sie erhalten eine Willkommens-E-Mail von NetSuite < billing@notification.netsuite.com > < system@sent-via.netsuite.com > mit Details wie Ihrem Benutzernamen und einem Link zum Einstellen Ihres Passworts.
- Verwenden Sie den Link Passwort festlegen, um das Passwort für den vom Anbieter bereitgestellten Benutzernamen festzulegen.

Eine NetSuite Oracle-Client-App und OAuth 2.0-Anmeldeinformationen erstellen

Um die Client ID und den Client Secret zu erhalten, erstellen Sie eine NetSuite Oracle-Client-App:

1. Loggen Sie sich über das NetSuite Kunden-Login in Ihr NetSuite Konto ein.
2. Wählen Sie „Einstellungen“ > „Unternehmen“ > „Funktionen aktivieren“.
3. Navigieren Sie zu dem SuiteCloudAbschnitt und aktivieren Sie das Kontrollkästchen REST WEB SERVICES unter SuiteTalk (Web Services).
4. Wählen Sie unter Authentifizierung verwalten das Kontrollkästchen OAUTH 2.0 aus. Klicken Sie auf Speichern.
5. Gehen Sie zu Setup > Integration > Integrationen verwalten und wählen Sie Neu, um eine OAuth2 2.0-Anwendung zu erstellen.
6. Geben Sie einen Namen Ihrer Wahl ein und lassen Sie den Status auf Aktiviert stehen.
7. Wenn diese Option aktiviert ist, deaktivieren Sie die Kontrollkästchen TBA: AUTHORIZATION FLOW und TOKEN-BASED AUTHENTICATION, die unter TOKEN-based Authentication angezeigt werden.
8. Wählen Sie unter 2.0 die Kontrollkästchen AUTHORIZATION CODE GRANT und PUBLIC CLIENT aus. OAuth
9. Notieren Sie sich unter Auth die Client-ID und das Client-Geheimnis.
10. Geben Sie eine REDIRECT-URI ein. Zum Beispiel https://us-east-1.console.aws.amazon.com/gluestudio/ oauth
11. Wählen Sie unter SCOPE das Kontrollkästchen REST WEB SERVICES aus.
12. Wählen Sie unter Benutzeranmeldedaten das Kontrollkästchen BENUTZERANMELDEINFORMATIONEN aus. Wählen Sie Save (Speichern) aus.
13. Notieren Sie sich das CONSUMER KEY/CLIENT ID and CONSUMER SECRET/CLIENT SECRET unter Client-Anmeldeinformationen. Diese Werte werden nur einmal angezeigt.

14 Erstellen Sie bei Bedarf eine ADMINISTRATOR-Rolle, indem Sie zu Benutzer/Rollen > Rollen verwalten > Neu navigieren.

15 Fügen Sie beim Erstellen einer benutzerdefinierten Rolle auf der Registerkarte „Berechtigungen“ Vollzugriff für die folgenden Entitäten/Funktionen hinzu:

- „Einzahlung“, „Artikel“, „Artikelabwicklung“, „Journaleintrag erstellen“, „Bestellung“, „Tochtergesellschaften“, „Lieferanten“, „Rechnungen“, „Autorisierung für Lieferantenrücksendungen“, „Zeit verfolgen“, „Kundenzahlung“, „Benutzerdefinierte Datensatzeinträge“, „Benutzerdefinierte Datensatztypen“, „REST Web Services“, „OAuth 2.0 Authorized Applications Management“, „Benutzerdefinierte Entitätsfelder“, „Mit OAuth 2.0-Zugriffstoken anmelden“.

Weitere Informationen finden Sie unter OAuth 2.0 in der NetSuite Applications Suite-Dokumentation.

Konfiguration von NetSuite Oracle-Verbindungen

Oracle NetSuite unterstützt den Grant-Typ AUTHORIZATION_CODE für. OAuth2 Die Art der Gewährung bestimmt, wie mit Oracle AWS Glue kommuniziert wird, um Zugriff NetSuite auf Ihre Daten anzufordern.

- Diese Art der Gewährung wird als „dreibeiniges“ angesehen, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App (AWS Glue verwaltete Client-Anwendung) verlassen, für die er außer der URL seiner NetSuite Oracle-Instanz keine OAuth zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Oracle weiter, NetSuite wo er sich anmelden und die angeforderten Berechtigungen für AWS Glue den Zugriff auf seine NetSuite Oracle-Instanz gewähren muss.
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Oracle zu erstellen NetSuite und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Oracle weitergeleitet, NetSuite um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.

- Öffentliche NetSuite Oracle-Dokumentation zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow finden Sie unter Öffentliche Apps.

So konfigurieren Sie eine NetSuite Oracle-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindung in erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Oracle aus NetSuite.
   b. Stellen Sie die NetSuite Oracle-Umgebung bereit.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus NetSuite Oracle-Entitäten lesen

Voraussetzung

Ein NetSuite Oracle-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie `deposit` oder`timebill`. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Order By	Unterstützt Limit	Unterstützt SELECT *	Unterstützt Partitionierung
Einzahlung	Ja	Nein	Ja	Ja	Ja
Beschreibung Artikel	Ja	Nein	Ja	Ja	Ja
Artikel aus dem Inventar	Ja	Nein	Ja	Ja	Ja
Versand von Artikeln	Ja	Nein	Ja	Ja	Ja
Artikelgruppe	Ja	Nein	Ja	Ja	Ja
Tagebuchintrag	Ja	Nein	Ja	Ja	Ja
Kaufartikel ohne Inventar	Ja	Nein	Ja	Ja	Ja
Wiederverkaufsartikel, der nicht auf Lager ist	Ja	Nein	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Order By	Unterstützt Limit	Unterstützt SELECT *	Unterstützt Partitionierung
Artikel zum Verkauf, der nicht auf Lager ist	Ja	Nein	Ja	Ja	Ja
Bestellungen	Ja	Nein	Ja	Ja	Ja
Tochtergesellschaft	Ja	Nein	Ja	Ja	Ja
Hersteller	Ja	Nein	Ja	Ja	Ja
Verkäufer Bill	Ja	Nein	Ja	Ja	Ja
Autorisierung der Rücksendung durch den Lieferanten	Ja	Nein	Ja	Ja	Ja
Zeitrechnung	Ja	Nein	Ja	Ja	Ja
Zahlung durch den Kunden	Ja	Nein	Ja	Ja	Ja
Anfrage zur Erfüllung	Ja	Nein	Ja	Ja	Ja
Item	Ja	Ja	Ja	Ja	Ja
Transaktionszeile	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Order By	Unterstützt Limit	Unterstützt SELECT *	Unterstützt Partitionierung
Buchhaltungszeile für Transaktionen	Ja	Ja	Ja	Ja	Ja
Benutzerdefinierte Datensatztypen (dynamisch)	Ja	Ja	Ja	Ja	Ja

Beispiel:

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1"
    }
)
```

Angaben zur NetSuite Oracle-Entität und zum Feld:

Oracle lädt die verfügbaren Felder NetSuite dynamisch unter der ausgewählten Entität. Je nach Datentyp des Felds unterstützt es die folgenden Filteroperatoren.

Datentyp des Felds	Unterstützte Filteroperatoren
String	WIE, =,! =
Datum	ZWISCHEN, =, <, <=, >, >=
DateTime	ZWISCHEN, <, <=, >, >=

Datentyp des Felds	Unterstützte Filteroperatoren
Numerischer Wert	=, !=, <, <=, >, >=
Boolesch	=, !=

Erwartetes Eingabeformat für boolesche Werte im Filterausdruck:

Entität	Boolesches Wertformat „wahr“	Boolesches Wertformat „falsch“	Beispiel
Entitäten vom Typ „Artikel“, „Transaktionsposition“, „Transaktionsbuchhaltungsposition“ und „Benutzerdefinierter Datensatz“	T oder t	F oder f	isinactive = „T“ oder isinactive = „t“
Alle anderen Entitäten	true	false	isinactive = wahr

Abfragen partitionieren

Feldbasierte Partitionierung

Der NetSuite Oracle-Konnektor verfügt über dynamische Metadaten, sodass unterstützte Felder für die feldbasierte Partitionierung dynamisch ausgewählt werden. Die feldbasierte Partitionierung wird für Felder mit dem Datentyp Integer BigInteger, Date oder unterstützt. DateTime

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`, und angeben`UPPER_BOUND`, `NUM_PARTITIONS` falls Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

Für das Zeitstempelfeld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

Beispiele für gültige Werte:

```
"TIMESTAMP \"1707256978123\""
"TIMESTAMP \"1702600882\""
"TIMESTAMP '2024-02-06T22:00:00:00.000Z'"
"TIMESTAMP '2024-02-06T22:00:00:00Z'"
"TIMESTAMP '2024-02-06'"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "id",
        "LOWER_BOUND": "1",
        "UPPER_BOUND": "10000",
        "NUM_PARTITIONS": "10"
    }
```

Datensatzbasierte Partitionierung

Sie können die zusätzliche Spark-Option bereitstellen, NUM_PARTITIONS wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Bei der datensatzbasierten Partitionierung wird die Gesamtzahl der vorhandenen Datensätze von der NetSuite Oracle-API abgefragt und durch die angegebene NUM_PARTITIONS Anzahl geteilt. Die resultierende Anzahl von Datensätzen wird dann gleichzeitig von jeder Unterabfrage abgerufen.

- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "3"
    }
```

NetSuite Oracle-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Oracle aufgeführt NetSuite:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name der NetSuite Oracle-Entität. Beispiel: Einzahlung.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Oracle NetSuite Rest API-Version, die Sie verwenden möchten. Der Wert wird v1 sein, da Oracle NetSuite derzeit nur Version v1 unterstützt.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Durch Kommas getrennte Liste der Spalten, die Sie für die ausgewählte Entität auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll (feldbasierte Partitionierung).
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes, der bei der feldbasierten Partitionierung verwendet wird.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes, der bei der feldbasierten Partitionierung verwendet wird.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen. Wird sowohl bei feld- als auch bei datensatzbasierter Partitionierung verwendet.
- `INSTANCEE_URL`(String) — Eine gültige NetSuite Instanz-URL mit Format https://{account-id}.suitetalk.api.netsuite.com.

Einschränkungen und Hinweise für Oracle NetSuite Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den NetSuite Oracle-Connector:

- Die Werte der Parameter access_token und refresh_token liegen im Format JSON Web Token (JWT) vor. Das Zugriffstoken ist 60 Minuten gültig, wohingegen das refresh_token sieben Tage gültig ist.
- Wenn Sie bei der Generierung von Client-ID und Client-Geheimnissen „PUBLIC CLIENT“ zusammen mit „AUTHORIZATION CODE GRANT“ auswählen, ist das Aktualisierungstoken nur drei Stunden gültig und darf nur einmal verwendet werden.
- Mit dem Connector können Sie maximal 1.000.000 Datensätze abrufen. Weitere Informationen finden Sie unter Ausführen von SuiteQL-Abfragen über REST-Webdienste.
- Partitionen werden so erstellt, dass jede Partition Datensätze in Vielfachen von 1000 abruft, außer möglicherweise die letzte, die die verbleibenden Datensätze abruft.
- Für Item-, Transaction Line- und Transaction Accounting Line-Objekte unterstützt der Konnektor einige Operatoren aus folgenden Gründen nicht:
  - Die Anwendung der `EQUAL_TO` `NOT_EQUAL_TO` Filteroperatoren auf Felder des Typs Datum führt zu unzuverlässigen Ergebnissen.
  - Die Anwendung des `LESS_THAN_OR_EQUAL_TO` Filteroperators auf Felder des Typs Datum führt zu unzuverlässigen Ergebnissen und verhält sich ähnlich wie der Operator. `LESS_THAN`
  - Die Anwendung des `GREATER_THAN` Filteroperators auf Felder des Typs Date= führt zu unzuverlässigen Ergebnissen und verhält sich ähnlich wie der Operator. `GREATER_THAN_OR_EQUAL_TO`
- Bei Objekten vom Typ Item, Transaction Line, Transaction Accounting Line und Custom Record Type werden boolesche Werte nicht im Standardformat, T/F sondern im Format angegeben, um die Konsistenz der Daten true/false. The connector maps the t/f values to true/false zu gewährleisten.

Verbindung zum OpenSearch Service herstellen in AWS Glue Studio

AWS Glue bietet integrierten Support für Amazon OpenSearch Service. AWS Glue Studio bietet eine visuelle Oberfläche, um eine Verbindung zu Amazon OpenSearch Service herzustellen, Datenintegrationsjobs zu erstellen und sie auf dem AWS Glue Studio serverlose Spark-Laufzeit. Diese Funktion ist nicht mit OpenSearch Service Serverless kompatibel.

AWS Glue Studio erstellt eine einheitliche Verbindung für Amazon OpenSearch Service. Weitere Informationen finden Sie unter Überlegungen.

Themen

- Eine OpenSearch Dienstverbindung erstellen
- Einen OpenSearch Dienstquellknoten erstellen
- Einen OpenSearch Service-Zielknoten erstellen
- Erweiterte Optionen

Eine OpenSearch Dienstverbindung erstellen

Voraussetzungen:

- Identifizieren Sie den Domain-Endpunkt *aosEndpoint* und den Port, von dem *aosPort* Sie lesen möchten, oder erstellen Sie die Ressource, indem Sie den Anweisungen in der Amazon OpenSearch Service-Dokumentation folgen. Weitere Informationen zum Erstellen einer Domain finden Sie unter Amazon OpenSearch Service-Domains erstellen und verwalten in der Amazon OpenSearch Service-Dokumentation.

  Ein Amazon OpenSearch Service-Domain-Endpunkt hat das folgende Standardformat: https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com. Weitere Informationen zur Identifizierung Ihres Domain-Endpunkts finden Sie unter Amazon OpenSearch Service-Domains erstellen und verwalten in der Amazon OpenSearch Service-Dokumentation.

  Identifizieren oder generieren Sie Anmeldeinformationen für die HTTP-Basisauthentifizierung *aosUser* und *aosPassword* für Ihre Domain.

Um eine Verbindung zum OpenSearch Service zu konfigurieren:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer OpenSearch Dienstanmeldedaten ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel USERNAME mit dem Wert. *aosUser*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel PASSWORD mit dem Wert. *aosPassword*

2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen,*connectionName*, für die future Verwendung in AWS Glue.

   - Wählen Sie bei der Auswahl eines Verbindungstyps OpenSearch Service aus.
   - Geben Sie bei der Auswahl eines Domänenendpunkts Folgendes an*aosEndpoint*.
   - Geben Sie bei der Auswahl eines Ports an*aosPort*.
   - Geben Sie bei der Auswahl eines AWS Secrets Folgendes an*secretName*.

Einen OpenSearch Dienstquellknoten erstellen

Voraussetzungen

- Eine mit einem AWS Secrets Manager geheimen Schlüssel konfigurierte AWS Glue OpenSearch Dienstverbindung, wie im vorherigen Abschnitt beschrieben,the section called “Eine OpenSearch Dienstverbindung erstellen”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Ein OpenSearch Dienstindex, aus dem Sie lesen möchten,*aosIndex*.

Eine OpenSearch Dienstdatenquelle hinzufügen

So fügen Sie eine Datenquelle — einen OpenSearch Dienstknoten hinzu:

1. Wählen Sie die Verbindung für Ihre OpenSearch Servicedatenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung herstellen müssen, wählen Sie OpenSearch Dienstverbindung erstellen. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Eine OpenSearch Dienstverbindung erstellen”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Geben Sie Index an, den Index, den Sie lesen möchten.
3. Geben Sie optional Query an, eine OpenSearch Abfrage, um spezifischere Ergebnisse zu liefern. Weitere Informationen zum Schreiben von OpenSearch Abfragen finden Sie unterthe section called “Aus dem OpenSearch Service lesen”.

4. Geben Sie in den Eigenschaften des benutzerdefinierten OpenSearch Dienstes nach Bedarf Parameter und Werte ein.

Einen OpenSearch Service-Zielknoten erstellen

Voraussetzungen

- Eine mit einem AWS Secrets Manager geheimen Schlüssel konfigurierte AWS Glue OpenSearch Dienstverbindung, wie im vorherigen Abschnitt beschrieben,the section called “Eine OpenSearch Dienstverbindung erstellen”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Ein OpenSearch Dienstindex, in den Sie schreiben möchten,*aosIndex*.

Ein OpenSearch Servicedatenziel hinzufügen

Um ein Datenziel — OpenSearch Serviceknoten hinzuzufügen:

1. Wählen Sie die Verbindung für Ihre OpenSearch Servicedatenquelle. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung herstellen müssen, wählen Sie OpenSearch Dienstverbindung erstellen. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Eine OpenSearch Dienstverbindung erstellen”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Geben Sie Index an, den Index, den Sie lesen möchten.
3. Geben Sie in den Eigenschaften des benutzerdefinierten OpenSearch Dienstes nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können erweiterte Optionen angeben, wenn Sie einen OpenSearch Service-Knoten erstellen. Diese Optionen sind dieselben wie bei der Programmierung AWS Glue für Spark-Skripte.

Siehe the section called “OpenSearch Serviceanschlüsse”.

Verbindung zu Okta herstellen

Die Okta-API ist die programmatische Schnittstelle zu Okta, die für die Verwaltung großer oder komplexer Okta-Konten und -Kampagnen verwendet wird. Wenn Sie ein Okta-Benutzer sind, können Sie eine Verbindung zu Ihrem Okta-Konto herstellen AWS Glue . Anschließend können Sie Okta als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Okta und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Okta
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Okta konfigurieren
- Konfiguration von Okta-Verbindungen
- Aus Okta-Entitäten lesen
- Referenz zu den Okta-Verbindungsoptionen
- Schritte zur Erstellung eines neuen Okta-Kontos und einer Entwickler-App
- Einschränkungen

AWS Glue Unterstützung für Okta

AWS Glue unterstützt Okta wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Okta abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Okta-API-Versionen

v1.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Okta konfigurieren

Bevor Sie Daten AWS Glue zu oder von Okta übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Okta-Konto. Weitere Informationen zum Erstellen eines Kontos finden Sie unterSchritte zur Erstellung eines neuen Okta-Kontos und einer Entwickler-App.
- Ihr Okta-Konto ist für den API-Zugriff aktiviert.
- Sie haben eine OAuth2 API-Integration in Ihrem Okta-Konto erstellt. Diese Integration stellt die Kundenanmeldedaten bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter Schritte zum Erstellen einer Client-App und unter OAuth2 2.0-Anmeldeinformationen: Neues Okta-Konto und Schritte zur Erstellung einer Entwickler-App
- Sie haben ein Okta-Konto mit einem. OktaApiToken Weitere Informationen finden Sie in der Okta-Dokumentation.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Okta-Konto herstellen. Für typische Verbindungen müssen Sie in Okta nichts weiter tun.

Konfiguration von Okta-Verbindungen

Okta unterstützt zwei Arten von Authentifizierungsmechanismen:

- OAuth auth: Okta unterstützt den `AUTHORIZATION_CODE` Grant-Typ.
  - Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Okta weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Okta-Instanz gewähren muss.
  - Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Okta zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Okta weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

  - Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
  - Weitere Informationen finden Sie in der öffentlichen Okta-Dokumentation zum Erstellen einer verbundenen App für den OAuth Autorisierungscodefluss.
- Benutzerdefinierte Authentifizierung:
  - Die öffentliche Okta-Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie in der Okta-Dokumentation.

So konfigurieren Sie eine Okta-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für die OAuth Authentifizierung:
      - Für vom Kunden verwaltete verbundene Apps — Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Für benutzerdefinierte Authentifizierung:
      - Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `OktaApiToken` Schlüssel enthalten.
2. Stellen Sie in AWS Glue Studio unter Datenverbindungen eine Verbindung her, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Verbindungen die Option Verbindung erstellen aus.
   b. Wählen Sie bei der Auswahl einer Datenquelle Okta aus.
   c. Geben Sie Ihre Okta-Subdomain an.
   d. Wählen Sie die Okta-Domain-URL Ihres Okta-Kontos aus.
   e. Wählen Sie die IAM-Rolle aus, die die AWS Glue folgenden Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

f. Wählen Sie den Authentifizierungstyp aus, um eine Verbindung zur Datenquelle herzustellen.

g. Geben Sie als OAuth2 Authentifizierungstyp die User Managed Client Application ClientId der Okta-App an.

h. Wählen Sie den aus`secretName`, den Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

i. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Okta-Entitäten lesen

Voraussetzungen

- Ein Okta-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Anwendungen	Ja	Ja	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Geräte	Ja	Ja	Nein	Ja	Ja
Gruppen	Ja	Ja	Ja	Ja	Ja
Benutzer	Ja	Ja	Ja	Ja	Ja
Benutzertypen	Nein	Nein	Nein	Ja	Nein

Beispiel

```
okta_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Okta",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "applications",
        "API_VERSION": "v1"
    }
```

Entitäts- und Felddetails von Okta

Liste der Entitäten:

- Anwendung: https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Application/
- Gerät: https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Device/
- Gruppe: https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Group/
- Benutzer: https://developer.okta.com/docs/api/openapi/okta-management/management/tag/User/
- Benutzertyp: https://developer.okta.com/docs/api/openapi/okta-management/management/tag/UserType/

Partitionierung von Abfragen

Zusätzliche Spark-OptionenPARTITION_FIELD,LOWER_BOUND,UPPER_BOUND, NUM_PARTITIONS können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:"2024-02-06".
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Beispiel

```
okta_read = glueContext.create_dynamic_frame.from_options(
    connection_type="okta",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "lastUpdated",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "lastMembershipUpdated"
        "LOWER_BOUND": "2022-08-10T10:28:46.000Z"
        "UPPER_BOUND": "2024-08-10T10:28:46.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Referenz zu den Okta-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Okta aufgeführt:

- ENTITY_NAME(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Okta.
- API_VERSION(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Okta Rest API-Version, die Sie verwenden möchten. Beispiel: v1.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.

- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Schritte zur Erstellung eines neuen Okta-Kontos und einer Entwickler-App

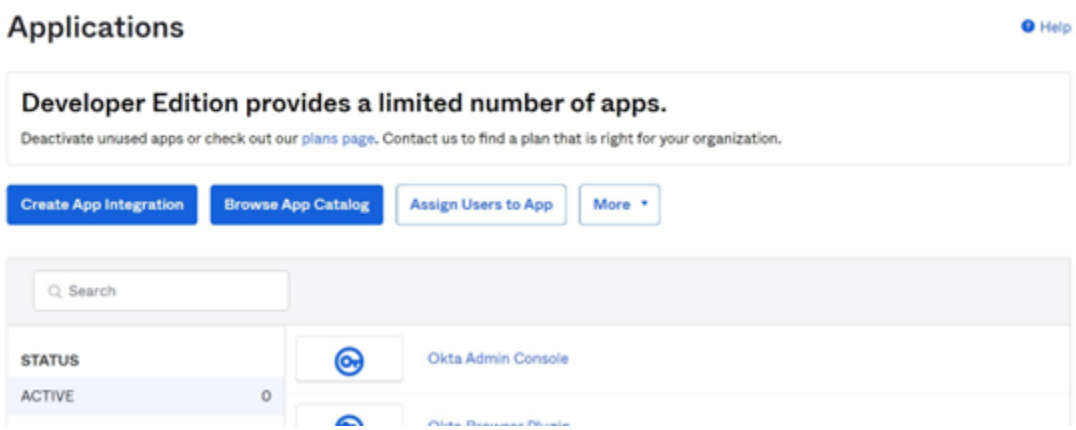
Erstellen Sie ein Entwicklerkonto bei Okta, um Zugriff auf die Okta-API zu erhalten. Ein kostenloses Okta-Entwicklerkonto bietet Zugriff auf die meisten wichtigen Entwicklerfunktionen, die für den Zugriff auf die Okt-API erforderlich sind.

Um ein Entwicklerkonto bei Okta zu erstellen

1. Navigieren Sie zu https://developer.okta.com/signup/.
2. Geben Sie die E-Mail-Adresse, den Vornamen, den Nachnamen und das Land/die Region für die Kontoinformationen ein. Wählen Sie Ich bin kein Roboter und dann Anmelden.
3. Eine Bestätigungsmail wird an Ihre registrierte E-Mail-ID gesendet. In Ihrer E-Mail erhalten Sie einen Link zur Aktivierung Ihres Okta-Entwicklerkontos. Wählen Sie Activate.
4. Sie werden zur Seite zum Zurücksetzen des Passworts weitergeleitet. Geben Sie das neue Passwort zweimal ein und wählen Sie Passwort zurücksetzen.
5. Sie werden zum Dashboard Ihres Okta-Entwicklerkontos weitergeleitet.

Um eine Client-App und OAuth 2.0-Anmeldeinformationen zu erstellen

1. Wählen Sie im Entwickler-Dashboard die Option App-Integration erstellen aus.



2. Das Fenster „Neue App-Integration erstellen“ wird angezeigt und enthält verschiedene Anmeldemethoden. Wählen Sie OIDC —OpenID Connect aus.
3. Scrollen Sie nach unten zum Abschnitt Anwendungstyp. Wählen Sie „Als Webanwendung“ und dann „Weiter“.
4. Geben Sie auf dem Bildschirm „Neue Web-App-Integration“ die folgenden Informationen ein:
   - Name der App-Integration — Geben Sie den Namen der App ein.
   - Art der Gewährung — Wählen Sie den Autorisierungscode und das Aktualisierungstoken aus der Liste aus.
   - Anmeldeweiterleitung URIs — Wählen Sie „URI hinzufügen“ und „Hinzufügen`https://{regioncode}.console.aws.amazon.com/appflow/oauth`“. Wenn Sie beispielsweise verwenden, können `us-west-2 (Oregon)` Sie hinzufügen`https://us-east-1.console.aws.amazon.com/appflow/oauth`.
   - Kontrollierter Zugriff — Ordnen Sie die App nach Bedarf Ihren Benutzergruppen zu und wählen Sie Speichern.
5. Ihre Kunden-ID und Ihr Kundengeheimnis werden generiert.

Einschränkungen

Die folgenden Einschränkungen gelten für den Okta-Connector:

- Für die Entität „Anwendungen“ kann nur ein Filter angewendet werden. Wenn mehr als ein Filter angewendet wird, werden 400 Bad Requests mit der Fehlerzusammenfassung „Ungültige Suchkriterien“ zurückgegeben.

- Die Option Sortieren nach kann nur bei Suchanfragen unterstützt werden. Beispielsweise `http://dev-15940405.okta.com/api/v1/groups?search=type e.q. "OKTA_GROUP"&sortBy=lastUpdated&sortOrder=asc`

Verbindung herstellen zu PayPal

PayPal ist ein Zahlungssystem, das Online-Geldtransfers zwischen Parteien erleichtert, z. B. Überweisungen zwischen Kunden und Online-Anbietern. Wenn Sie ein PayPal Benutzer sind, enthält Ihr Konto Daten über Ihre Transaktionen, z. B. deren Zahler, Daten und Status. Sie können AWS Glue es verwenden, um Daten von bestimmten AWS Diensten oder anderen unterstützten Anwendungen PayPal zu übertragen.

Themen

- AWS Glue Unterstützung für PayPal
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration PayPal
- PayPal Verbindungen konfigurieren
- Aus PayPal Entitäten lesen
- PayPal Verbindungsoptionen
- Einschränkungen und Hinweise für PayPal Connector

AWS Glue Unterstützung für PayPal

AWS Glue unterstützt PayPal wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen PayPal.

Wird es als Ziel unterstützt?

Nein.

Unterstützte PayPal API-Versionen

Die folgenden PayPal API-Versionen werden unterstützt:

- v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
     ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben

Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration PayPal

Bevor Sie Daten von AWS Glue übertragen können PayPal, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein PayPal Konto mit Kundenanmeldedaten.
- Ihr PayPal Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem PayPal Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun PayPal.

PayPal Verbindungen konfigurieren

PayPal unterstützt den Gewährungstyp CLIENT CREDENTIALS für OAuth2.

- Dieser Gewährungstyp gilt als 2-Legged OAuth 2.0, da er von Clients verwendet wird, um außerhalb des Benutzerkontextes ein Zugriffstoken zu erhalten. AWS Glue ist in der Lage, die Client-ID und das Client-Geheimnis zur Authentifizierung zu verwenden PayPal APIs , die von benutzerdefinierten Diensten bereitgestellt werden, die Sie definieren.
- Jeder benutzerdefinierte Dienst gehört einem Benutzer, der nur über eine API verfügt und über eine Reihe von Rollen und Berechtigungen verfügt, die den Dienst autorisieren, bestimmte Aktionen auszuführen. Ein Zugriffstoken ist einem einzelnen benutzerdefinierten Dienst zugeordnet.
- Dieser Gewährungstyp führt zu einem Zugriffstoken, das nur von kurzer Dauer ist und durch erneutes Aufrufen des `/v2/oauth2/token` Endpunkts erneuert werden kann.
- Öffentliche PayPal Dokumentation für OAuth 2.0 mit Client-Anmeldeinformationen finden Sie unter Authentifizierung.

So konfigurieren Sie eine PayPal Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps PayPal.
   b. Geben Sie die PayPal Instanz `INSTANCE_URL` an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Anmeldeinformationen für OAuth Version 2.0 erhalten

Um die Rest-API aufzurufen, müssen Sie Ihre Client-ID und Ihr Client-Geheimnis gegen ein Zugriffstoken eintauschen. Weitere Informationen finden Sie unter Erste Schritte mit PayPal REST APIs.

Aus PayPal Entitäten lesen

Voraussetzung

Ein PayPal Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen,`transaction`.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Transaktion	Ja	Ja	Nein	Ja	Ja

Beispiel:

```
paypal_read = glueContext.create_dynamic_frame.from_options(
    connection_type="paypal",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "transaction",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://api-m.paypal.com"
    }
```

PayPal Entitäts- und Felddetails:

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Transaktion	Datum der Transaktionseinleitung	DateTime	Between

Entität	Feld	Datentyp	Unterstützte Operatoren
	Datum und Uhrzeit der letzten Aktualisierung	String	N/A
	Art des Zahlungsinstruments	String	=
	balance_affecting_records_only	String	=
	speicher-ID	String	=
	Terminal-ID	String	=
	Transaktionswährung	String	=
	transaction_id	String	N/A
	Transaktionsstatus	String	N/A
	transaction_type	String	N/A
	Transaktionsinformationen	Struct	N/A
	zahler_info	Struct	N/A
	Versandinformationen	Struct	N/A
	Informationen zum Einkaufswagen	Struct	N/A
	Informationen speichern	Struct	N/A
	Informationen zur Auktion	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Informationen zu Anreizen	Struct	N/A

Partitionierung von Abfragen

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir den Wert im ISO-Format.

  Beispiele für gültige Werte:

  ```
  "2024-07-01T00:00:00.000Z"
  ```
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Das folgende Feld wird für die entitätsweise Partitionierung unterstützt:

Entity-Name	Partitionierung von Feldern	Datentyp
Transaktion	Datum der Transaktionseinlei tung	DateTime

Beispiel:

```
paypal_read = glueContext.create_dynamic_frame.from_options(
    connection_type="paypal",
```

```
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "transaction",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "transaction_initiation_date"
        "LOWER_BOUND": "2024-07-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-07-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

PayPal Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für PayPal:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in PayPal.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. PayPal Rest-API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.

Einschränkungen und Hinweise für PayPal Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den PayPal Konnektor:

- In der PayPal Transaktionsdokumentation wird erwähnt, dass es maximal drei Stunden dauert, bis ausgeführte Transaktionen im Listentransaktionsaufruf erscheinen. Es wurde jedoch beobachtet, dass es je nach Zeit länger dauert `last_refreshed_datetime`. Hier `last_refreshed_datetime` ist der Zeitraum, bis zu dem Ihnen Daten vom zur Verfügung stehen APIs.
- Wenn der kleiner als der angeforderte `end_date` Wert `last_refreshed_datetime` ist, `end_date` entspricht der dem, `last_refreshed_datetime` da wir nur Daten bis zu diesem Zeitpunkt haben.
- Das `transaction_initiation_date` Feld ist ein obligatorischer Filter, der für die `transaction` Entität bereitgestellt werden muss, und der maximal unterstützte Datumsbereich für dieses Feld beträgt 31 Tage.
- Wenn Sie die `transaction` Entitäts-API-Anfrage mit anderen Filtern (Abfrageparametern) als dem `transaction_initiation_date` Feld aufrufen, wird erwartet, dass der Wert des `ending_balance`Felds in der Antwort nicht abgerufen wird.

Verbindung zu Pendo herstellen

Pendo bietet einen umfangreichen Datenspeicher für Benutzerinteraktionsdaten. Kunden übertragen diese Daten an, AWS sodass sie sie mit anderen Produktdaten verknüpfen, zusätzliche Analysen und Dashboards durchführen und Benachrichtigungen einrichten können, wenn sie dies wünschen.

Themen

- AWS Glue Unterstützung für Pendo
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Pendo konfigurieren
- Pendo-Verbindungen konfigurieren
- Aus Pendo-Entitäten lesen
- Pendo-Verbindungsoptionen
- Einschränkungen

AWS Glue Unterstützung für Pendo

AWS Glue unterstützt Pendo wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Pendo abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Pendo API-Versionen

v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Pendo konfigurieren

Bevor Sie die Übertragung von AWS Glue Pendo verwenden können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein Pendo-Konto mit aktiviertem `apiKey` WITH`write access`.
- Ihr Pendo-Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem Pendo-Konto verbinden AWS Glue . Für typische Verbindungen müssen Sie in Pendo nichts weiter tun.

Pendo-Verbindungen konfigurieren

Pendo unterstützt benutzerdefinierte Authentifizierung.

Die öffentliche Pendo-Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie unter Authentifizierung — Pendo REST API-Dokumentation

So konfigurieren Sie eine Pendo-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Für vom Kunden verwaltete verbundene App — Secret sollte die verbundene App Consumer Secret mit `apiKey` als Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, für jede Verbindung in ein Secret zu erstellen AWS Glue.

2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Pendo aus.
   b. Geben Sie die Adresse `instanceUrl` der Pendo-Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
    "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "secretsmanager:DescribeSecret",
                    "secretsmanager:GetSecretValue",
                    "secretsmanager:PutSecretValue",
                    "ec2:CreateNetworkInterface",
                    "ec2:DescribeNetworkInterface",
                    "ec2:DeleteNetworkInterface",
                ],
                "Resource": "*"
            }
        ]
    }
```

   d. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Pendo-Entitäten lesen

Voraussetzungen

Ein Pendo-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

- Merkmal
- Leitfaden
- Seite
- Bericht
- Daten melden
- Besucher
- Account
- Veranstaltung
- Hauptereignis
- Veranstaltung leiten
- Seite „Ereignis“
- Umfrage-Ereignis
- Ereignis verfolgen

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Funktion	Nein	Nein	Nein	Ja	Nein
Richtlinie	Nein	Nein	Nein	Ja	Nein
Seite	Nein	Nein	Nein	Ja	Nein
Bericht	Nein	Nein	Nein	Ja	Nein
Daten melden	Nein	Nein	Nein	Ja	Nein
Besucher (Aggregations-API)	Ja	Nein	Ja	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Konto (Aggregations-API)	Ja	Nein	Ja	Ja	Nein
Ereignis (Aggregations-API)	Ja	Nein	Ja	Ja	Nein
Funktionsereignis (Aggregations-API)	Ja	Nein	Ja	Ja	Ja
Leitfadenereignis (Aggregations-API)	Ja	Nein	Ja	Ja	Ja
Konto (Aggregations-API)	Ja	Nein	Ja	Ja	Ja
Seitenereignis (Aggregation-API)	Ja	Nein	Ja	Ja	Ja
Umfrageereignis (Aggregations-API)	Ja	Nein	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Ereignis verfolgen (Aggregations-API)	Ja	Nein	Ja	Ja	Ja

Beispiel

```
Pendo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.Pendo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "feature",
        "API_VERSION": "v1",
        "INSTANCE_URL": "instanceUrl"
    }
```

Partitionierung von Abfragen

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,LOWER_BOUND, und angebenUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das DateTime Feld akzeptieren wir den Wert im ISO-Format.

  Beispiel für einen gültigen Wert:

  ```
  "2024-07-01T00:00:00.000Z"
  ```
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

In der folgenden Tabelle werden die Informationen zur Unterstützung von Feldern zur Partitionierung von Entitäten beschrieben:

Entity-Name
Ereignis
Funktionsereignis
Veranstaltung leiten
Seite „Ereignis“
Umfrage-Ereignis
Ereignis verfolgen

Beispiel:

```
pendo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.pendo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "event",
        "API_VERSION": "v1",
        "INSTANCE_URL": "instanceUrl"
        "NUM_PARTITIONS": "10",
        "PARTITION_FIELD": "appId"
        "LOWER_BOUND": "4656"
        "UPPER_BOUND": "7788"
    }
```

Pendo-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Pendo aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Pendo.
- `INSTANCE_URL`(Zeichenfolge) — (Erforderlich) Eine gültige Pendo-Instanz-URL mit den folgenden zulässigen Werten:

  - Standard
  - Europa
  - US1
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. Pendo Engage Rest API-Version, die Sie verwenden möchten. Zum Beispiel: 3.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Pendo-Anschluss:

- Die Paginierung wird in Pendo nicht unterstützt.
- Die Filterung wird nur von den Aggregate-API-Objekten (`Account`,`Event`,`Feature Event`,`Guide Events`, `Page Event` `Poll Event` `Track Event`, und) unterstützt `Visitor`
- DateTimeRange ist ein obligatorischer Filterparameter für Aggregate-API-Objekte (`Event`Feature Event`,`Guide Events`,,`Page Event`, `Poll Event`,`Track Event`)
- Der DayRange-Zeitraum wird auf den Beginn des Zeitraums in der Zeitzone abgerundet. Wenn zum Beispiel ein Filter angegeben ist, `2023-01-12T07:55:27.065Z` dann wird dieser Zeitraum auf den Anfang der Periode gerundet, das heißt`2023-01-12T00:00:00Z`.

Verbindung zu Pipedrive herstellen

Pipedrive ist ein CRM für Vertriebspipelines, das kleine Unternehmen dabei unterstützt, Leads zu verwalten, Vertriebsaktivitäten nachzuverfolgen und mehr Geschäfte abzuschließen. Pipedrive ermöglicht es Vertriebsteams in kleinen Unternehmen: Prozesse zu rationalisieren und Vertriebsdaten in einem einheitlichen CRM-Vertriebstool zu konsolidieren. Wenn Sie ein Pipedrive-Nutzer sind, können Sie sich mit Ihrem Pipedrive-Account verbinden AWS Glue . Anschließend können Sie Pipedrive als Datenquelle in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Pipedrive und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Pipedrive
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Pipedrive konfigurieren
- Konfiguration von Pipedrive-Verbindungen
- Aus Pipedrive-Entitäten lesen
- Referenz zu den Pipedrive-Verbindungsoptionen
- Einschränkungen

AWS Glue Unterstützung für Pipedrive

AWS Glue unterstützt Pipedrive wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Pipedrive abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Pipedrive-API-Versionen

v1.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Pipedrive konfigurieren

Bevor Sie Daten von Pipedrive AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben einen Pipedrive-Account.
- Ihr Pipedrive-Konto ist für den API-Zugriff aktiviert.

Wenn Sie diese Anforderungen erfüllen, sind Sie bereit, sich mit Ihrem AWS Glue Pipedrive-Account zu verbinden. Für typische Verbindungen müssen Sie in Pipedrive nichts weiter tun.

Konfiguration von Pipedrive-Verbindungen

Pipedrive unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2

- Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App verlassen, in der er außer der Pipedrive-Instance-URL keine OAuth zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Pipedrive weiter, wo er sich anmelden und den angeforderten Berechtigungen für den Zugriff auf seine Pipedrive-Instanz gewähren AWS Glue muss.
- Benutzer sollten sich dafür entscheiden, ihre eigene verbundene App in Pipedrive zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Pipedrive weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen zu autorisieren AWS Glue .
- Dieser Grant-Typ führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist eine Stunde lang aktiv und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Weitere Informationen finden Sie in der Dokumentation zum Erstellen einer verbundenen App für den AUTHORIZATION_CODE-Flow OAuth .

So konfigurieren Sie eine Pipedrive-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps sollte Secret das Consumer Secret der verbundenen App mit USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET als Schlüssel enthalten.
2. Stellen Sie unter Datenverbindungen eine AWS Glue Studio Verbindung her, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Datenverbindungen die Option Verbindung erstellen aus.
   b. Wählen Sie bei der Auswahl einer Datenquelle Pipedrive aus.
   c. Geben Sie Ihre Pipedrive-Instance-URL an.
   d. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   e. Stellen Sie die benutzerverwaltete Client-Anwendung ClientId des Pipedrive bereit, mit dem Sie eine Verbindung herstellen möchten.
   f. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte. Wählen Sie Weiter.

4. Geben Sie den Verbindungsnamen ein und wählen Sie Weiter.
5. Wählen Sie auf der nächsten Seite Verbindung erstellen. Sie werden aufgefordert, sich bei Pipedrive anzumelden. Geben Sie Ihren Benutzernamen und Ihr Passwort ein und wählen Sie Anmelden.
6. Sobald Sie angemeldet sind, wählen Sie Weiter zur App. Jetzt ist Ihre Verbindung einsatzbereit.
7. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Pipedrive-Entitäten lesen

Voraussetzungen

- Ein Pipedrive-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Aktivitäten	Ja	Ja	Nein	Ja	Ja
Aktivitätstyp	Nein	Nein	Nein	Ja	Nein
Anruflisten	Nein	Nein	Nein	Ja	Nein
Währungen	Ja	Ja	Nein	Ja	Nein
Angebote	Ja	Ja	Ja	Ja	Ja
Führt	Ja	Ja	Ja	Ja	Nein
Leitquellen	Nein	Ja	Nein	Ja	Nein
Etiketten für Blei	Nein	Nein	Nein	Nein	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Hinweise	Ja	Ja	Ja	Ja	Ja
Organisation	Ja	Ja	Nein	Ja	Ja
Berechtigungssätze	Ja	Nein	Nein	Ja	Nein
Personen	Ja	Ja	Ja	Ja	Ja
Pipelines	Nein	Ja	Nein	Ja	Nein
Produkte	Ja	Ja	Nein	Ja	Ja
Rollen	Nein	Ja	Nein	Ja	Nein
Stufen	Ja	Ja	Nein	Ja	Nein
Benutzer	Nein	Nein	Nein	Ja	Nein

Beispiel

```
pipedrive_read= glueContext.create_dynamic_frame.from_options(
    connection_type="PIPEDRIVE",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "activites",
        "API_VERSION": "v1"
    }
```

Pipedrive-Entitäts- und Felddetails

Liste der Entitäten:

- Aktivitäten: https://developers.pipedrive.com/docs/api/v1/Activities
- Art der Aktivität: https://developers.pipedrive.com/docs/api/v1/ActivityTypes
- Anruflisten: https://developers.pipedrive.com/docs/api/v1/CallLogs

- Währungen: https://developers.pipedrive.com/docs/api/v1/Currencies
- Angebote: https://developers.pipedrive.com/docs/api/v1/Deals
- Führt: https://developers.pipedrive.com/docs/api/v1/Leads
- Leitquellen: https://developers.pipedrive.com/docs/api/v1/LeadSources
- Bleietiketten: https://developers.pipedrive.com/docs/api/v1/LeadLabels
- Hinweise: https://developers.pipedrive.com/docs/api/v1/Notes
- Organizations: https://developers.pipedrive.com/docs/api/v1/Organizations
- Berechtigungssätze: https://developers.pipedrive.com/docs/api/v1/PermissionSets
- Personen: https://developers.pipedrive.com/docs/api/v1/Persons
- Rohrleitungen: https://developers.pipedrive.com/docs/api/v1/Pipelines
- Produkte: https://developers.pipedrive.com/docs/api/v1/Products
- Rollen: https://developers.pipedrive.com/docs/api/v1/Roles
- Etappen: https://developers.pipedrive.com/docs/api/v1/Stages
- Nutzer: https://developers.pipedrive.com/docs/api/v1/Users

Entität	Datentyp	Unterstützte Operatoren
Aktivitäten, Angebote, Notizen, Organisation, Personen und Produkte.	Datum	'='
	Ganzzahl	'='
	String	'='
	Boolesch	'='

Abfragen partitionieren

In Pipedrive unterstützt nur ein Feld (due_date) aus der Entität Activities die feldbasierte Partitionierung. Es ist ein Datumsfeld.

Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern

würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:"2024-02-06".
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.

Beispiel

```
pipedrive_read = glueContext.create_dynamic_frame.from_options(
    connection_type="PIPEDRIVE",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "activites",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "due_date"
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z"
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Referenz zu den Pipedrive-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Pipedrive aufgeführt:

- ENTITY_NAME(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Pipedrive.
- API_VERSION(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Die Pipedrive Rest API-Version, die Sie verwenden möchten. Beispiel: v1.
- INSTANCE_URL(String) — (Erforderlich) URL der Instanz, in der der Benutzer die Operationen ausführen möchte. Beispiel: v1.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.

- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Pipedrive-Konnektor:

- Pipedrive unterstützt die feldbasierte Partitionierung nur für eine Entität (Aktivitäten).
- Pipedrive unterstützt die datensatzbasierte Partitionierung für Aktivitäten, Geschäfte, Notizen, Personen, Organizations und Produkte.
- In der Entität Deals gibt das Statusfeld als Filter den gesamten Datensatz zurück, wenn ein ungültiger Wertefilterwert verwendet wird.
- In der Entität Deals wird die Bestellung mit mehreren Feldern nicht unterstützt.
- Um Leistungsdaten zu erhalten, verwenden wir ein lokales AWS Konto. Aufgrund der Beschränkung, das Zugriffstoken lokal zu aktualisieren, schlägt der AWS Glue Job zur Verarbeitung von 1 GB Daten jedoch fehl. Folglich haben wir den Leistungstest mit 179 MB Daten optimiert, und die obigen Ergebnisse basieren auf dieser Optimierung. Dennoch haben wir beobachtet, dass der SaaS-Endpunkt mit zunehmender Anzahl von Partitionen mehr Zeit in Anspruch nimmt als eine einzelne Partition. Wir haben uns bezüglich dieses Verhaltens mit dem Pipedrive-Support-Team beraten und sie haben uns darüber informiert, dass Pipedrive Anfragen stillschweigend drosselt und die Antwort verzögert. Wenn Sie den AWS Glue Job mit großen Datensätzen ausführen oder denselben API-Endpunkt mehrmals aufrufen, kann es daher aufgrund der Implementierung der Pipedrive-API zu einem Timeout-Problem kommen. Die Antwortzeiten von Connector und Shim sinken jedoch erwartungsgemäß mit zunehmender Anzahl von Partitionen.

Verbindung zum Productboard herstellen

Productboard ist das Produktmanagementsystem, das Produktteams dabei unterstützt, die richtigen Produkte schneller auf den Markt zu bringen. Über 3.000 moderne, produktorientierte Unternehmen wie Zendesk und Microsoft nutzen Productboard UiPath, um zu verstehen, was Benutzer wirklich benötigen, zu priorisieren, was als Nächstes gebaut werden soll, und um alle für ihre Roadmap zu gewinnen.

Themen

- AWS Glue Unterstützung für Productboard
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Productboard konfigurieren
- Konfiguration von Productboard-Verbindungen
- Aus Productboard-Entitäten lesen
- Verbindungsoptionen für das Productboard
- Einen Productboard-Account erstellen
- Einschränkungen

AWS Glue Unterstützung für Productboard

AWS Glue unterstützt Productboard wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Productboard abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Productboard-API-Versionen

v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Productboard konfigurieren

Bevor Sie die Übertragung von Productboard verwenden AWS Glue können, müssen Sie die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Sie haben ein Productboard-Konto mit E-Mail und Passwort. Weitere Informationen zur Erstellung eines Accounts findest du unter Einen Productboard-Account erstellen.
- Sie müssen AWS ein Konto mit dem Servicezugriff auf AWS Glue erstellt haben.
- Sie haben die Authentifizierungsdetails eines Productboard-Kontos — entweder JWT-Token, wenn Sie Custom Auth verwenden möchten, oder Client-ID und Secret, wenn Sie .0 verwenden möchten. OAuth2
- Wenn der Benutzer es verwenden möchte`OAuth2.0`, registrieren Sie Ihre Anwendung bei Productboard und richten Sie die Anwendung ein, indem Sie den Anweisungen unter So integrieren Sie Productboard via — Entwicklerdokumentation folgen. OAuth2

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem AWS Glue Productboard-Konto verbinden. Für typische Verbindungen müssen Sie in Productboard nichts weiter tun.

Konfiguration von Productboard-Verbindungen

Productboard unterstützt benutzerdefinierte Authentifizierung und. `OAuth2.0` Denn `OAuth2.0` Productboard unterstützt den `AUTHORIZATION_CODE` Zuschusstyp.

- Dieser Grant-Typ wird als „dreibeiniges“ angesehen, `OAuth` da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App verlassen, für die er außer seiner Productboard Client ID und Client Secret keine `OAuth` weiteren Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Productboard weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Productboard-Instanz gewähren muss.
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Productboard zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu

Productboard weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Eine öffentliche Productboard-Dokumentation zur Erstellung einer verbundenen App für `AUTHORIZATION_CODE OAuth` Flow finden Sie unter So integrieren Sie sich mit Productboard via OAuth2 — Entwicklerdokumentation.

So konfigurieren Sie eine Productboard-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Für `OAuth` Auth — Für vom Kunden verwaltete verbundene App: Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   - Für `Custom auth` — Für vom Kunden verwaltete verbundene Apps: Secret sollte die verbundene App `JWT token` mit einem `access_token` Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, für jede Verbindung ein Geheimnis zu erstellen AWS Glue.

2. Erstellen Sie unter Datenverbindungen eine Verbindung AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl einer Datenquelle Productboard aus.
   b. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
```

```
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

c. Wählen Sie den Authentifizierungstyp aus, um eine Verbindung zur Datenquelle herzustellen:

- Für `OAuth` die Authentifizierung — Geben Sie das `Token URL` und `User Managed Client Application ClientId` der Productboard-App an.

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Productboard-Entitäten lesen

Voraussetzungen

Ein Productboard-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

- Missbrauchsberichte
- Automation
- Kampagnen
- Klicken Sie auf Details
- Listen
- Mitglieder
- Details öffnen
- Segmente

- Geschäfte
- Abgemeldet

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Features	Ja	Ja	Nein	Ja	Ja
Komponenten	Nein	Ja	Nein	Ja	Nein
Produkte	Nein	Ja	Nein	Ja	Nein
Status der Funktionen	Nein	Ja	Nein	Ja	Ja
Definitionen benutzerdefinierter Felder	Nein	Ja	Nein	Ja	Nein
Werte für benutzerdefinierte Felder	Ja	Ja	Nein	Ja	Nein

Beispiel

```
Productboard_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Productboard",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "feature",
        "API_VERSION": "1"
    }
```

Details zur Entität und zum Feld im Productboard

- Funktionen
- Komponenten
- Status der Funktionen
- Produkte
- Definitionen von benutzerdefinierten Feldern
- Werte für benutzerdefinierte Felder

Verbindungsoptionen für das Productboard

Im Folgenden sind die Verbindungsoptionen für Productboard aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Productboard.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Productboard Engage Rest API-Version, die Sie verwenden möchten. Zum Beispiel: 3.0.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Einen Productboard-Account erstellen

1. Gehen Sie zur Productboard-Anmeldeseite, geben Sie Ihre E-Mail-ID und Ihr Passwort ein und wählen Sie dann Log me in.
2. Geben Sie im Feld Accountname den Namen Ihres Productboard-Kontos ein und aktivieren Sie dann das Kontrollkästchen Ich stimme den Datenschutzbestimmungen zu.
3. Geben Sie auf der Seite Jetzt deinen Workspace erstellen im Feld Workspace-URL die URL für deinen neuen Workspace ein. Wähle dann Weiter, um zur nächsten Seite zu gelangen und die restlichen Details anzugeben.

   Dadurch wird Ihr Testkonto erstellt. Das Testkonto ist 15 Tage lang kostenlos. Nach Ablauf der Testphase können Sie einen kostenpflichtigen Tarif erwerben. Notiere dir deine E-Mail-Adresse,

dein Passwort und deine Workspace-URL. Sie benötigen diese Informationen, um in future auf Ihr Konto zugreifen zu können.“

Registrierung einer **`OAuth2.0`** Anwendung

1. Gehen Sie zur Productboard-Anmeldeseite, geben Sie Ihre E-Mail-ID und Ihr Passwort ein und wählen Sie Anmelden.
2. Wählen Sie das Benutzersymbol in der oberen rechten Ecke und wählen Sie dann im Drop-down-Menü Konto und Abrechnung aus.
3. Wählen Sie Extras und dann Registrierte Apps aus dem Drop-down-Menü aus.
4. Suchen Sie nach einer App und wählen Sie „App registrieren“.
5. Geben Sie die folgenden Details ein:
   - App-Name — Name der App.
   - Firma/ Organisation — Name Ihres Unternehmens oder Ihrer Organisation.
   - App-Website — Website der App.
   - Umleitungs-URI — Ein Umleitungs-URI-Muster ist ein URI-Pfad (oder eine durch Kommas getrennte Liste von Pfaden), zu dem Productboard (falls angefordert) weiterleiten kann, wenn der Anmeldevorgang abgeschlossen ist. Beispiel: `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`
6. Wählen Sie Create (Erstellen) aus.
7. Die Client-ID und das Client-Geheimnis werden jetzt sichtbar sein. Kopieren Sie sie und speichern Sie sie an einem sicheren Ort. Wählen Sie dann „Fertig“.

> **Note**
>
> Ihre Client-ID- und Client Secret-Zeichenfolgen sind Anmeldeinformationen, die verwendet werden, um eine Verbindung mit diesem Connector herzustellen, wenn Sie AppFlow oder verwenden AWS Glue.

Anmeldeinformationen werden abgerufen CustomAuth

1. Gehen Sie zur Productboard-Anmeldeseite, geben Sie Ihre E-Mail-ID und Ihr Passwort ein und wählen Sie Anmelden.

Sie werden auf die Startseite weitergeleitet.

2. Navigieren Sie auf der Startseite zu Workspace-Einstellungen > Integrationen > Öffentlich APIs > Zugriffstoken.

> **Note**
>
> Wenn der APIs Bereich „Öffentlich“ nicht sichtbar ist, ist dein Konto möglicherweise Teil des Essentials-Plans. Für den Zugriff auf API-Token ist mindestens ein Pro-Plan erforderlich. Die Funktionen und Namen des Plans können sich ändern. Weitere Informationen zu den Paketen finden Sie unter Productboard-Preise.

3. Wählen Sie +, um ein neues Token zu generieren, und stellen Sie sicher, dass Sie es sicher speichern, damit Sie später darauf future können.

`OAuth2.0`Anmeldeinformationen werden erstellt

Um die `OAuth2.0` Authentifizierung mit dem Productboard-Connector nutzen zu können, müssen Sie Ihre Anwendung auf der Productboard-Plattform registrieren und ein `Client ID` und generieren. `Client Secret`

1. Navigieren Sie zur Productboard-Anmeldeseite, geben Sie Ihre E-Mail-ID und Ihr Passwort ein und wählen Sie Anmelden.
2. Um eine neue OAuth2 Anwendung mit Ihrem Productboard-Konto zu registrieren, navigieren Sie zur Producboard-Seite.
3. Füllen Sie die erforderlichen Felder aus und wählen Sie die erforderlichen Bereiche für jede Entität aus, auf die Sie zugreifen möchten.

> **Note**
>
> Sie haben die folgenden vier Bereiche ausgewählt, die für die sechs unterstützten Entitäten erforderlich sind.

4. Die Weiterleitungs-URL muss das folgende Format haben: `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`

> **Note**
>
> Die Appflow-Weiterleitungen können URLs sich ändern. Sobald sie verfügbar ist, aktualisieren Sie bitte die Weiterleitung URLs für die AWS Glue Plattform.

5. Die Client-ID und der geheime Client-Schlüssel sind jetzt sichtbar. Kopieren Sie sie und speichern Sie sie an einem sicheren Ort.
6. Sie können sie einrichten und überprüfen, `OAuth2` indem Sie den Schritten in der OAuth2 Entwicklerdokumentation zur Integration mit Productboard folgen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Productboard-Connector:

- Productboard unterstützt weder feldbasierte noch datensatzbasierte Partitionierung.

Verbindung herstellen zu QuickBooks

QuickBooks ist eine führende Buchhaltungsanwendung für kleine und mittlere Unternehmen. QuickBooks Buchhaltungsanwendungen stammen aus den 1980er Jahren als eines der ersten Produkte von Intuit und waren daher ursprünglich Desktop-Software. Heute QuickBooks bietet es mehrere Buchhaltungs- und Geschäftsfinanzanwendungen sowohl als installierbare Software als auch als Cloud-basierte SaaS-Software an. Als QuickBooks Benutzer können Sie eine Verbindung AWS Glue zu Ihrem QuickBooks Konto herstellen. Anschließend können Sie es QuickBooks als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten QuickBooks zwischen AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für QuickBooks
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration QuickBooks
- QuickBooks Verbindungen konfigurieren
- Aus QuickBooks Entitäten lesen
- QuickBooks Verbindungsoptionen
- Einschränkungen und Hinweise für QuickBooks Connector

AWS Glue Unterstützung für QuickBooks

AWS Glue unterstützt QuickBooks wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen QuickBooks.

Wird es als Ziel unterstützt?

Nein.

Unterstützte QuickBooks API-Versionen

Die folgenden QuickBooks API-Versionen werden unterstützt:

- v3

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
     "Statement": [
      {
             "Effect": "Allow",
             "Action": [
                 "glue:ListConnectionTypes",
                 "glue:DescribeConnectionType",
                 "glue:RefreshOAuth2Tokens",
                 "glue:ListEntities",
                 "glue:DescribeEntity"
             ],
             "Resource": "*"
         }
```

```
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration QuickBooks

Bevor Sie Daten von AWS Glue übertragen können QuickBooks, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein QuickBooks Konto.
- Ihr QuickBooks Konto ist für den API-Zugriff aktiviert.

Weitere Informationen finden Sie in der QuickBooks Dokumentation zu den folgenden Themen:

- Erstellen Sie ein Intuit-Konto
- Erstelle deine App und beginne mit der Entwicklung

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem QuickBooks Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun QuickBooks.

QuickBooks Verbindungen konfigurieren

QuickBooks unterstützt den Grant-Typ AUTHORIZATION_CODE für. OAuth2 Der Grant-Typ bestimmt, wie mit Ihnen AWS Glue kommuniziert wird, um Zugriff QuickBooks auf Ihre Daten anzufordern.

- Diese Art der Gewährung wird als „dreibeiniges“ angesehen, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in zu erstellen QuickBooks und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin weitergeleitet, QuickBooks um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
- Öffentliche QuickBooks Dokumentation zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow finden Sie unter Setup OAuth 2.0.

So konfigurieren Sie eine QuickBooks Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps QuickBooks.
   b. Geben Sie die Instanz-URL und die Firmen-ID der QuickBooks Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus QuickBooks Entitäten lesen

Voraussetzung

Ein QuickBooks Objekt, aus dem Sie lesen möchten.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Account	Ja	Ja	Ja	Ja	Ja
Bill	Ja	Ja	Ja	Ja	Ja
Informationen zum Unternehmen	Nein	Nein	Nein	Ja	Nein
Customer	Ja	Ja	Ja	Ja	Ja
Mitarbeiter	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Schätzung	Ja	Ja	Ja	Ja	Ja
Rechnung	Ja	Ja	Ja	Ja	Ja
Item	Ja	Ja	Ja	Ja	Ja
Payment (Zahlung)	Ja	Ja	Ja	Ja	Ja
Präferenzen	Nein	Nein	Nein	Ja	Nein
Gewinn und Verlust	Ja	Nein	Nein	Ja	Nein
Steuerbehörde	Ja	Ja	Ja	Ja	Ja
Anbieter	Ja	Ja	Ja	Ja	Ja

Beispiel:

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v3"
    }
```

QuickBooks Entitäts- und Felddetails:

Weitere Informationen zu den Entitäten und Felddetails finden Sie unter:

- Account
- Rechnung
- CompanyInfo

- Kunde
- Mitarbeiter
- Schätzung
- Rechnung
- Item
- Bezahlung
- Präferenzen
- ProfitAndLoss
- TaxAgency
- Vendor

Abfragen partitionieren

Feldbasierte Partitionierung:

In unterstützen die QuickBooks Felder Integer und DateTime Datatype feldbasierte Partitionierung.

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS falls Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

  ```
  "2024-05-07T02:03:00.00Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "12345678690123456789",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v3",
        "PARTITION_FIELD": "MetaData_CreateTime"
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z"
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Datensatzbasierte Partitionierung:

Die ursprüngliche Abfrage ist in NUM_PARTITIONS mehrere Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können:

- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "1234567890123456789",
        "ENTITY_NAME": "Bill",
        "API_VERSION": "v3",
        "NUM_PARTITIONS": "10"
    }
```

QuickBooks Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für QuickBooks:

- ENTITY_NAME(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in QuickBooks.
- INSTANCE_URL(String) — (Erforderlich) Eine gültige QuickBooks Instanz-URL.

- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen verwendet. QuickBooks Rest-API-Version, die Sie verwenden möchten.
- `REALM_ID`(Zeichenfolge) — Eine ID, die ein einzelnes QuickBooks Online-Unternehmen identifiziert, an das Sie Anfragen senden.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen und Hinweise für QuickBooks Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den QuickBooks Konnektor:

- In der `taxAgency` API funktioniert die Reihenfolge nach Filtern nicht wie erwartet.

Verbindung zu Salesforce herstellen

Salesforce bietet Software für das Kundenbeziehungsmanagement (CRM), die Sie bei Vertrieb, Kundenservice, E-Commerce und mehr unterstützt. Wenn Sie ein Salesforce-Benutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce-Konto herstellen. Anschließend können Sie Salesforce als Datenquelle oder Ziel in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Salesforce und AWS Services oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Salesforce
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Salesforce
- Wenden Sie das Systemadministrator-Profil an
- Konfiguration von Salesforce-Verbindungen
- Aus Salesforce lesen
- An Salesforce schreiben
- Salesforce-Verbindungsoptionen
- Einschränkungen für den Salesforce-Connector
- Richten Sie den Autorisierungscode-Flow für Salesforce ein
- Richten Sie den OAuth JWT-Bearer-Flow für Salesforce ein

AWS Glue Unterstützung für Salesforce

AWS Glue unterstützt Salesforce wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Salesforce abzufragen.

Als Ziel unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Datensätze in Salesforce zu schreiben.

Unterstützte Salesforce-API-Versionen

Die folgenden Salesforce-API-Versionen werden unterstützt

- v58.0
- v59.0
- v60.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispiel-IAM-Richtlinie werden die erforderlichen Berechtigungen für die Erstellung, Verwaltung und Verwendung von Salesforce-Verbindungen innerhalb von AWS Glue ETL-Jobs beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie

die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Wenn Sie beim Erstellen einer Salesforce-Verbindung Netzwerkoptionen angeben, müssen die folgenden Aktionen ebenfalls in der IAM-Rolle enthalten sein:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

Informationen zu Salesforce-Verbindungen ohne ETL finden Sie unter Voraussetzungen für Zero-ETL.

Informationen zu Salesforce-Verbindungen ohne ETL finden Sie unter Voraussetzungen für Zero-ETL.

Konfiguration von Salesforce

Bevor Sie Daten AWS Glue zu oder von Salesforce übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Salesforce-Konto.
- Ihr Salesforce-Konto ist für den API-Zugriff aktiviert. Der API-Zugriff ist standardmäßig für die Enterprise, Unlimited, Developer und Performance Editionen aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce-Konto herstellen. AWS Glue bewältigt die verbleibenden Anforderungen mit der AWS verwalteten verbundenen Anwendung.

Die AWS verwaltete verbundene Anwendung für Salesforce

Mit der AWS verwalteten verbundenen Anwendung können Sie in weniger Schritten eine Salesforce-Verbindung herstellen. In Salesforce ist eine verbundene Anwendung ein Framework, das externen Anwendungen beispielsweise AWS Glue den Zugriff auf Ihre Salesforce-Daten mithilfe von OAuth 2.0 autorisiert. Um die AWS verwaltete verbundene Anwendung zu verwenden, erstellen Sie mithilfe der Konsole eine Salesforce-Verbindung AWS Glue . Wenn Sie die Verbindung konfigurieren, legen Sie den OAuth Gewährungstyp auf Autorisierungscode fest und lassen Sie das Kontrollkästchen AWS Verwaltete Client-Anwendung verwenden aktiviert.

Beim Speichern der Verbindung werden Sie zu Salesforce weitergeleitet, um sich anzumelden und den AWS Glue Zugriff auf Ihr Salesforce-Konto zu genehmigen.

Wenden Sie das Systemadministratorprofil an

Gehen Sie in Salesforce wie folgt vor, um das Systemadministratorprofil anzuwenden:

1. Navigieren Sie in Salesforce zu Einstellungen > Verbundene Anwendungen > OAuth Nutzung verbundener Apps.
2. Suchen Sie in der Liste der verbundenen Anwendungen nach Installieren AWS Glue und wählen Sie diese aus. Wählen Sie bei Bedarf Entsperren aus.
3. Navigiere zu Einstellungen > Verbundene Apps verwalten und wähle dann AWS Glue. Wählen Sie unter OAuth Richtlinien die Option Admin approved users are pre-authorized aus und wählen Sie das Profil Systemadministrator aus. Diese Aktion beschränkt den Zugriff AWS Glue nur auf Benutzer mit dem Systemadministratorprofil.

Wenden Sie das Systemadministrator-Profil an

Gehen Sie in Salesforce wie folgt vor, um das Systemadministratorprofil anzuwenden:

1. Navigieren Sie in Salesforce zu Einstellungen > Verbundene Anwendungen > OAuth Nutzung verbundener Apps.
2. Suchen Sie in der Liste der verbundenen Anwendungen nach Installieren AWS Glue und wählen Sie diese aus. Wählen Sie bei Bedarf Entsperren aus.

3. Navigiere zu Einstellungen > Verbundene Apps verwalten und wähle dann AWS Glue. Wählen Sie unter OAuth Richtlinien die Option Admin approved users are pre-authorized aus und wählen Sie das Profil Systemadministrator aus. Diese Aktion beschränkt den Zugriff AWS Glue nur auf Benutzer mit dem Systemadministratorprofil.

Konfiguration von Salesforce-Verbindungen

So konfigurieren Sie eine Salesforce-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für den Grant-Typ JWT_TOKEN sollte das Geheimnis den JWT_TOKEN-Schlüssel mit seinem Wert enthalten.
   b. AuthorizationCode Für den Grant-Typ:
      i. Für eine AWS verwaltete verbundene App muss ein leeres Geheimnis oder ein Geheimnis mit einem temporären Wert angegeben werden.
      ii. Bei einer vom Kunden verwalteten verbundenen App sollte das Geheimnis die verbundene App `Consumer Secret` mit `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` dem Schlüssel enthalten.
   c. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindung in erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Salesforce aus.
   b. Geben Sie die INSTANCE_URL der Salesforce-Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Geben Sie die Salesforce-Umgebung an.
   d. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
```

```
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

e. Wählen Sie den OAuth2 Grant-Typ aus, den Sie für die Verbindungen verwenden möchten. Die Art der Gewährung bestimmt, wie mit Salesforce AWS Glue kommuniziert wird, um Zugriff auf Ihre Daten anzufordern. Ihre Wahl wirkt sich auf die Anforderungen aus, die Sie erfüllen müssen, bevor Sie die Verbindung herstellen. Sie können einen der folgenden Typen wählen:

- JWT_BEARER Grant-Typ: Dieser Grant-Typ eignet sich gut für Automatisierungsszenarien, da er es ermöglicht, vorab ein JSON Web Token (JWT) mit den Berechtigungen eines bestimmten Benutzers in der Salesforce-Instanz zu erstellen. Der Ersteller hat die Kontrolle darüber, wie lange das JWT gültig ist. AWS Glue kann das JWT verwenden, um ein Zugriffstoken zu erhalten, das zum Aufrufen von Salesforce verwendet wird. APIs

  Dieser Ablauf setzt voraus, dass der Benutzer in seiner Salesforce-Instanz eine verbundene Anwendung erstellt hat, die die Ausgabe von JWT-basierten Zugriffstoken für Benutzer ermöglicht.

  Informationen zum Erstellen einer verbundenen Anwendung für den OAuth JWT-Bearer-Flow finden Sie unter OAuth 2.0 JWT-Bearer-Flow zur Integration. server-to-server Informationen zum Einrichten des JWT-Bearer-Flows mit der verbundenen Salesforce-Anwendung finden Sie unter. Richten Sie den OAuth JWT-Bearer-Flow für Salesforce ein
- AUTHORIZATION_CODE Gewährungstyp: Dieser Gewährungstyp wird als „dreibeiniges“ eingestuft, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue verbundene Anwendung (AWS Glue verwaltete Client-Anwendung) verlassen, für die er außer der URL seiner Salesforce-Instanz keine OAuth zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Salesforce weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Salesforce-Instanz gewähren muss.

Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene Anwendung in Salesforce zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Client-ID und ihren eigenen geheimen Client-Schlüssel anzugeben. In diesem Szenario werden sie weiterhin zu Salesforce weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.

Informationen zum Erstellen einer verbundenen App für den OAuth Autorisierungscode-Flow finden Sie unterthe section called “Richten Sie den Autorisierungscode-Flow für Salesforce ein”.

f. Wählen Sie `secretName` die aus, in AWS Glue der Sie für diese Verbindung die OAuth 2.0-Token speichern möchten.

g. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

4. Wenn Sie Netzwerkoptionen bereitstellen, gewähren Sie der IAM-Rolle auch die folgenden Berechtigungen:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

Konfiguration von Salesforce-Verbindungen mit der AWS CLI

Sie können Salesforce-Verbindungen mit der AWS CLI erstellen:

```
aws glue create-connection --connection-input \
"{\"Name\": \"salesforce-conn1\",\"ConnectionType\": \"SALESFORCE\",
\"ConnectionProperties\": {\"ROLE_ARN\": \"arn:aws:iam::123456789012:role/glue-role\",
\"INSTANCE_URL\": \"https://example.my.salesforce.com\"},\"ValidateCredentials\": true,
\"AuthenticationConfiguration\": {\"AuthenticationType\": \"OAUTH2\",\"SecretArn\":
 \"arn:aws:secretsmanager:us-east-1:123456789012:secret:salesforce-conn1-secret-IAmcdk
\",\"OAuth2Properties\": {\"OAuth2GrantType\": \"JWT_BEARER\",\"TokenUrl\": \"https://
login.salesforce.com/services/oauth2/token\"}}}" \
--endpoint-url https://glue.us-east-1.amazonaws.com \
--region us-east-1
```

Aus Salesforce lesen

Voraussetzung

Ein Salesforce-sObject, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie `Account` oder oder`Case`. `Opportunity`

Beispiel:

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0"
    }
```

Abfragen partitionieren

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`, und angeben`UPPER_BOUND`, `NUM_PARTITIONS` ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für Datums- oder Timestamp-Felder akzeptiert der Konnektor das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

Beispiele für gültige Werte:

```
"TIMESTAMP \"1707256978123\""
"TIMESTAMP '2018-01-01 00:00:00.000 UTC'"
"TIMESTAMP \"2018-01-01 00:00:00 Pacific/Tahiti\""
"TIMESTAMP \"2018-01-01 00:00:00\""
"TIMESTAMP \"-123456789\" Pacific/Tahiti"
"TIMESTAMP \"1702600882\""
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.
- TRANSFER_MODE: unterstützt zwei Modi: SYNC undASYNC. Der Standardwert ist SYNC. Wenn diese Option auf gesetzt istASYNC, wird die Bulk-API 2.0-Abfrage für die Verarbeitung verwendet.

Beispiel:

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "PARTITION_FIELD": "SystemModstamp",
        "LOWER_BOUND": "TIMESTAMP '2021-01-01 00:00:00 Pacific/Tahiti'",
        "UPPER_BOUND": "TIMESTAMP '2023-01-10 00:00:00 Pacific/Tahiti'",
        "NUM_PARTITIONS": "10",
        "TRANSFER_MODE": "ASYNC"
    }
```

An Salesforce schreiben

Voraussetzungen

Ein Salesforce-sObject, in das Sie schreiben möchten. Sie benötigen den Objektnamen wie `Account` oder oder`Case`. `Opportunity`

Der Salesforce-Connector unterstützt vier Schreibvorgänge:

- INSERT

- UPSERT
- AKTUALISIERUNG
- DELETE

Wenn `UPSERT` Sie den Schreibvorgang verwenden, muss die `ID_FIELD_NAMES` Option bereitgestellt werden, um das externe ID-Feld für die Datensätze anzugeben.

Sie können auch Verbindungsoptionen hinzufügen:

- `TRANSFER_MODE`: Unterstützt zwei Modi: `SYNC` und`ASYNC`. Der Standardwert ist `SYNC`. Wenn diese Option auf gesetzt ist`ASYNC`, wird Bulk API 2.0 Ingest für die Verarbeitung verwendet.
- `FAIL_ON_FIRST_ERROR`: Der Standardwert ist`FALSE`, was bedeutet, dass der AWS Glue Job weiterhin alle Daten verarbeitet, auch wenn es einige fehlerhafte Schreibdatensätze gibt. Wenn dieser Wert auf gesetzt ist`TRUE`, schlägt der AWS Glue Job fehl, falls es Datensätze gibt, die beim Schreiben fehlschlagen konnten, und die Verarbeitung wird nicht fortgesetzt.

Beispiel

```
salesforce_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "WRITE_OPERATION": "INSERT",
        "TRANSFER_MODE": "ASYNC",
        "FAIL_ON_FIRST_ERROR":""true"
    }
```

Salesforce-Verbindungsoptionen

Die folgenden Verbindungsoptionen werden für den Salesforce-Connector unterstützt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Salesforce.
- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Salesforce Rest API-Version, die Sie verwenden möchten.

- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.

  Bei der Bereitstellung eines Filterprädikats wird nur der AND Operator unterstützt. Andere Operatoren wie OR und IN werden derzeit nicht unterstützt.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- UPPER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- IMPORT_DELETED_RECORDS(Zeichenfolge) — Standard: FALSE. Wird zum Lesen verwendet. Um die gelöschten Datensätze während der Abfrage abzurufen.
- WRITE_OPERATION(Zeichenfolge) — Standard: INSERT. Wird zum Schreiben verwendet. Der Wert sollte INSERT, UPDATE, UPSERT, DELETE sein.
- ID_FIELD_NAMES(Zeichenfolge) — Standard: null. Erforderlich für UPDATE und UPSERT.

Einschränkungen für den Salesforce-Connector

Die folgenden Einschränkungen gelten für den Salesforce-Connector:

- Wir unterstützen nur Spark SQL und Salesforce SOQL wird nicht unterstützt.
- Auftragslesezeichen werden nicht unterstützt.
- Bei Salesforce-Feldnamen wird zwischen Groß- und Kleinschreibung unterschieden. Beim Schreiben in Salesforce müssen die Daten der Groß-/Kleinschreibung der in Salesforce definierten Felder entsprechen.

Richten Sie den Autorisierungscode-Flow für Salesforce ein

Informationen zur Aktivierung des OAuth 2.0-Autorisierungscode-Flows finden Sie in der öffentlichen Salesforce-Dokumentation.

So konfigurieren Sie die verbundene Anwendung:

1. Aktivieren Sie das Kontrollkästchen OAuth „Einstellungen aktivieren“.
2. Geben Sie im Textfeld Callback-URL eine oder mehrere Weiterleitungen URLs für AWS Glue ein.

   Weiterleitungen URLs haben das folgende Format:

   https://*region*.console.aws.amazon. com/gluestudio/oauth

   In dieser URL ist Region der Code für die AWS Region, in die AWS Glue Sie Daten von Salesforce übertragen. Der Code für die Region USA Ost (Nord-Virginia) lautet beispielsweise`us-east-1`. Für diese Region lautet die URL wie folgt:

   https://us-east-1.console.aws.amazon.com/gluestudio/oauth

   Informationen zu den AWS Glue unterstützten AWS Regionen und ihren Codes finden Sie in der allgemeinen Referenz unter AWS Glue Endpunkte und KontingenteAWS .
3. Aktivieren Sie das Kontrollkästchen „Secret für Web Server Flow erforderlich“.
4. Fügen Sie in der Liste Verfügbare OAuth Bereiche die folgenden Bereiche hinzu:
   - Benutzerdaten verwalten über APIs (API)
   - Greifen Sie auf benutzerdefinierte Berechtigungen zu (custom_permissions)
   - Greifen Sie auf den IdentitätsURL-Dienst zu (ID, Profil, E-Mail, Adresse, Telefon)
   - Greifen Sie auf eindeutige Benutzerkennungen (Openid) zu
   - Anfragen können jederzeit ausgeführt werden (refresh_token, offline_access)
5. Stellen Sie die Aktualisierungstoken-Richtlinie für die verbundene App so ein, dass das Aktualisierungstoken bis zum Widerruf gültig ist. Andernfalls schlagen Ihre Jobs fehl, wenn Ihr Aktualisierungstoken abläuft. Weitere Informationen zum Überprüfen und Bearbeiten der Aktualisierungstoken-Richtlinie finden Sie unter Verwalten von OAuth Zugriffsrichtlinien für eine verbundene Anwendung in der Salesforce-Dokumentation.

Richten Sie den OAuth JWT-Bearer-Flow für Salesforce ein

Informationen zur Aktivierung der server-to-server Integration mit OAuth 2.0-JSON-Web-Tokens finden Sie in der öffentlichen Salesforce-Dokumentation.

Nachdem Sie ein JWT erstellt und die verbundene Anwendung in Salesforce entsprechend konfiguriert haben, können Sie eine neue Salesforce-Verbindung mit dem in Ihrem Secrets Manager Secret festgelegten `JWT_TOKEN` Schlüssel erstellen. Stellen Sie beim Erstellen der Verbindung den OAuth Gewährungstyp auf JWT Bearer Token ein.

Verbindung zur Salesforce Marketing Cloud herstellen

Salesforce Marketing Cloud ist ein Anbieter von Marketingautomatisierungs- und Analysesoftware für E-Mail-, Mobil-, Social- und Online-Marketing. Es bietet auch Beratungs- und Implementierungsdienstleistungen an. Als Salesforce Marketing Cloud-Benutzer können Sie eine Verbindung AWS Glue zu Ihrem Salesforce Marketing Cloud-Konto herstellen. Anschließend können Sie Salesforce Marketing Cloud als Datenquelle oder Ziel in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Salesforce Marketing Cloud und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Salesforce Marketing Cloud
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration der Salesforce Marketing Cloud
- Konfiguration von Salesforce Marketing Cloud-Verbindungen
- Aus Salesforce Marketing Cloud-Entitäten lesen
- Verbindungsoptionen für die Salesforce Marketing Cloud
- Einschränkungen und Hinweise für den Salesforce Marketing Cloud Connector

AWS Glue Unterstützung für Salesforce Marketing Cloud

AWS Glue unterstützt Salesforce Marketing Cloud wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus der Salesforce Marketing Cloud abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Salesforce Marketing Cloud-API-Versionen

Die folgenden Salesforce Marketing Cloud-API-Versionen werden unterstützt:

- v1

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration der Salesforce Marketing Cloud

Bevor Sie Daten aus der Salesforce Marketing Cloud übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Salesforce Marketing Cloud-Konto. Weitere Informationen finden Sie unter Ein Salesforce Marketing Cloud-Konto erstellen.
- Ihr Salesforce Marketing Cloud-Konto ist für den API-Zugriff aktiviert. Der API-Zugriff ist standardmäßig für die Enterprise, Unlimited, Developer und Performance Editionen aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce Marketing Cloud-Konto herstellen. Für typische Verbindungen müssen Sie in der Salesforce Marketing Cloud nichts weiter tun.

Ein Salesforce Marketing Cloud-Konto erstellen

Für die Salesforce Marketing Cloud müssen Sie sich für die Kontoerstellung an den Anbieter wenden. Wenn Sie oder Ihr Unternehmen eine Verbindung zu Salesforce haben, wenden Sie sich an Ihren Salesforce-Kundenbetreuer, um eine Salesforce Marketing Cloud-Lizenz anzufordern. Andernfalls können Sie wie folgt Kontakt von einem Salesforce-Mitarbeiter anfordern:

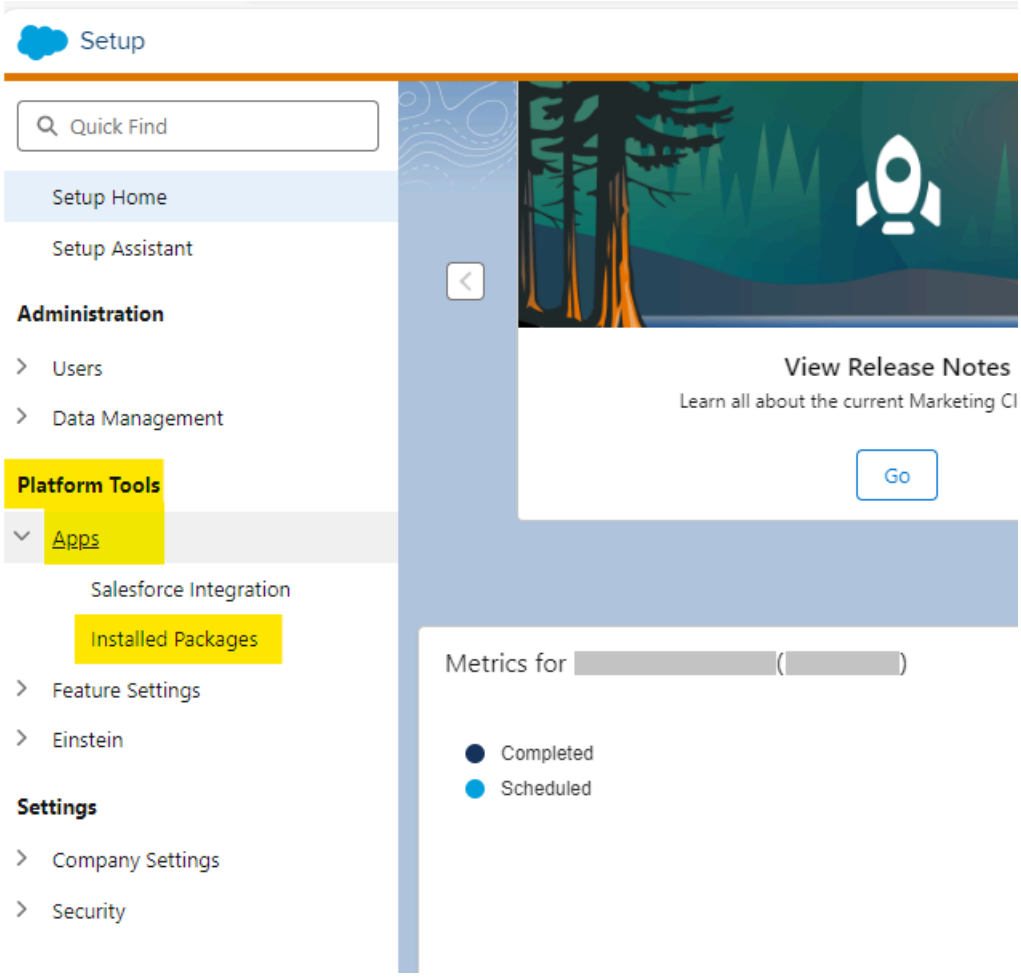
1. Gehen Sie zu https://www.salesforce.com/in/products/marketing-cloud/overview/und wählen Sie Anmelden aus.
2. Wählen Sie oben rechts auf der Seite den Link Kontaktieren Sie uns aus.
3. Geben Sie die erforderlichen Informationen in das Formular ein und wählen Sie „Kontaktiere mich“.

Ein Salesforce-Mitarbeiter wird sich mit Ihnen in Verbindung setzen, um Ihre Anforderungen zu besprechen.

Ein Projekt und OAuth 2.0-Anmeldeinformationen erstellen

Um ein Projekt und OAuth 2.0-Anmeldeinformationen zu erhalten:

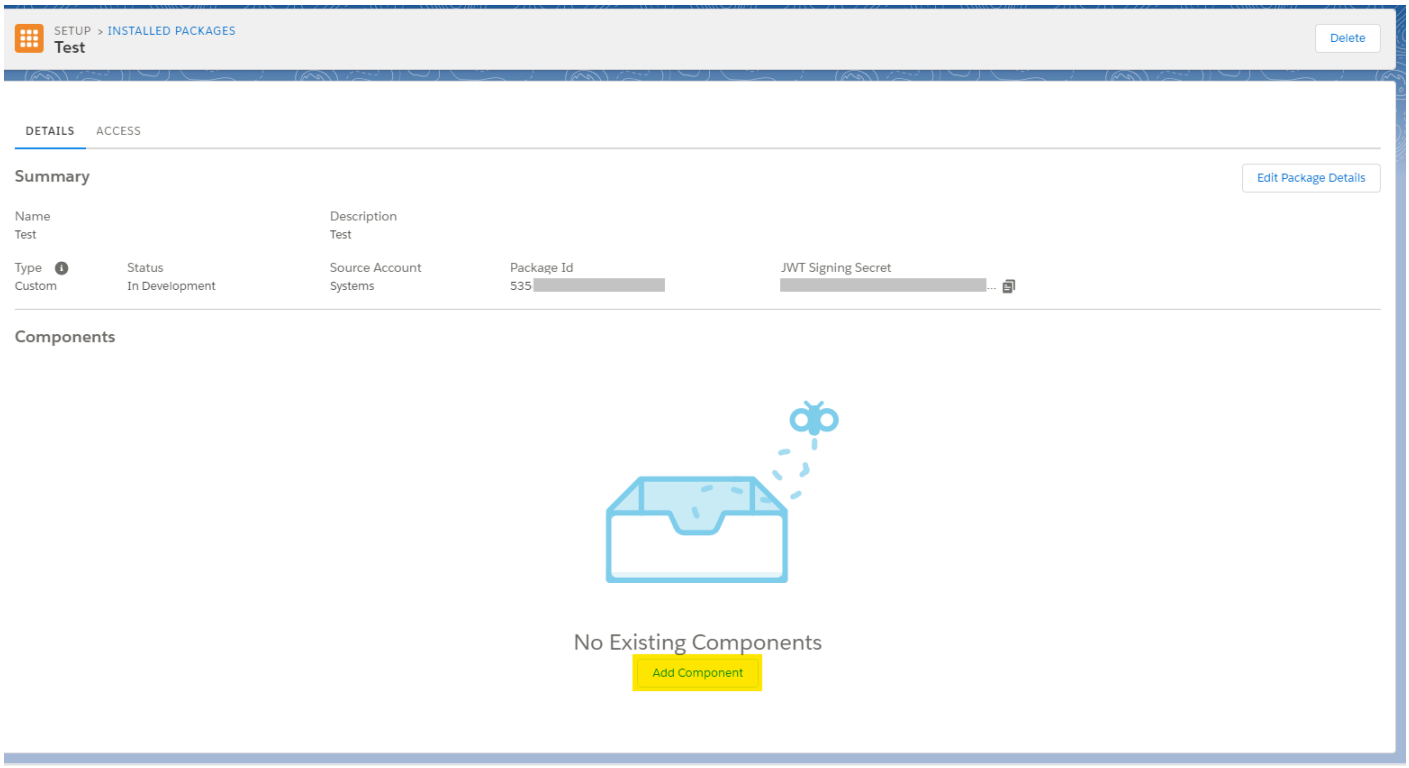
1. Melden Sie sich mit Ihrem Benutzernamen und Passwort bei Ihrer Salesforce Marketing Cloud-Instanz an und authentifizieren Sie sich mit Ihrer registrierten Handynummer.
2. Klicken Sie oben rechts auf Ihr Profil und gehen Sie dann zu Setup.
3. Wählen Sie unter Plattformtools die Option Apps und dann Installierte Pakete aus.



4. Klicken Sie auf der Seite Installierte Pakete oben rechts auf Neu. Geben Sie den Namen und die Beschreibung des Pakets an.

   Speichern Sie das Paket. Nachdem das Paket gespeichert wurde, können Sie die Paketdetails anzeigen.

5. Wählen Sie auf der Detailseite des Pakets im Abschnitt Komponente die Option Komponente hinzufügen aus.



6. Wählen Sie als Komponententyp „API-Integration“ aus und klicken Sie auf Weiter.
7. Wählen Sie als Integrationstyp „Server-to-Server“ (mit dem OAuth Gewährungstyp Client-Anmeldeinformationen) aus und klicken Sie auf Weiter.
8. Fügen Sie die Bereiche entsprechend Ihren Anforderungen hinzu und klicken Sie auf Speichern.

Konfiguration von Salesforce Marketing Cloud-Verbindungen

Salesforce Marketing Cloud unterstützt den Zuschusstyp CLIENT CREDENTIALS für OAuth2.

- Dieser Gewährungstyp gilt als 2-Legged OAuth 2.0, da er von Kunden verwendet wird, um außerhalb des Benutzerkontextes ein Zugriffstoken zu erhalten. AWS Glue ist in der Lage, die Client-ID und das Client-Geheimnis zur Authentifizierung der Salesforce Marketing Cloud zu verwenden APIs , die von benutzerdefinierten Diensten bereitgestellt werden, die Sie definieren.
- Jeder benutzerdefinierte Service gehört einem Nur-API-Benutzer, der über eine Reihe von Rollen und Berechtigungen verfügt, die den Service autorisieren, bestimmte Aktionen auszuführen. Ein Zugriffstoken ist einem einzelnen benutzerdefinierten Dienst zugeordnet.

- Dieser Gewährungstyp führt zu einem Zugriffstoken, das nur von kurzer Dauer ist und durch Aufrufen eines Identitätsendpunkts erneuert werden kann.
- Die öffentliche Salesforce Marketing Cloud-Dokumentation für OAuth 2.0 mit Kundenanmeldedaten finden Sie unter Einrichten Ihrer Entwicklungsumgebung für erweiterte Pakete.

So konfigurieren Sie eine Salesforce Marketing Cloud-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen einen geheimen Schlüssel für Ihre Verbindungen in erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Salesforce Marketing Cloud aus.
   b. Geben Sie `Subdomain Endpoint` die Salesforce Marketing Cloud an, mit der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Salesforce Marketing Cloud-Entitäten lesen

Voraussetzung

Ein Salesforce Marketing Cloud-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie `Activity` oder`Campaigns`. Die folgende Tabelle zeigt die unterstützten Entitäten.

Unterstützte Entitäten für die Quelle:

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt SELECT *	Unterstützt Partitionierung
Rückruf bei Ereignisbenachrichtigung	REST	Nein	Nein	Nein	Ja	Nein
Seed-Liste	REST	Nein	Ja	Nein	Ja	Nein
Aufstellen	REST	Ja	Ja	Nein	Ja	Nein
Domänenverifizierung	REST	Ja	Ja	Ja	Ja	Nein
Verschachtelte Tags für Objekte	REST	Ja	Nein	Nein	Ja	Nein
Kontakt	REST	Nein	Ja	Nein	Ja	Nein
Abonnement für Ereignisb	REST	Nein	Nein	Nein	Ja	Nein

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt SELECT *	Unterstützt Partitionierung
enachrichtigungen						
Messaging	REST	Nein	Ja	Nein	Ja	Nein
Aktivität	SOAP	Nein	Nein	Nein	Ja	Ja
Bounce-Ereignis	SOAP	Nein	Nein	Nein	Ja	Ja
Klicken Sie auf Event	SOAP	Nein	Nein	Nein	Ja	Ja
Inhaltsbereich	SOAP	Nein	Nein	Nein	Ja	Ja
Datenerweiterung	SOAP	Nein	Ja	Nein	Ja	Ja
Email	SOAP	Nein	Ja	Nein	Ja	Ja
Ereignis mit weitergeleiteter E-Mail	SOAP	Nein	Ja	Nein	Ja	Ja
E-Mail weiterleiten OptInEvent	SOAP	Nein	Ja	Nein	Ja	Ja
Link	SOAP	Nein	Ja	Nein	Ja	Ja
Link Senden	SOAP	Nein	Ja	Nein	Ja	Ja

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt SELECT *	Unterstützt Partitionierung
Auflisten	SOAP	Nein	Ja	Nein	Ja	Ja
Abonnent auflisten	SOAP	Nein	Ja	Nein	Ja	Ja
Ereignis nicht gesendet	SOAP	Nein	Ja	Nein	Ja	Ja
Veranstaltung öffnen	SOAP	Nein	Ja	Nein	Ja	Ja
Senden	SOAP	Nein	Ja	Nein	Ja	Ja
Ereignis gesendet	SOAP	Nein	Ja	Nein	Ja	Ja
Subscriber	SOAP	Nein	Ja	Nein	Ja	Ja
Ereignis der Umfrage	SOAP	Nein	Ja	Nein	Ja	Ja
Unsub-Ereignis	SOAP	Nein	Ja	Nein	Ja	Ja
Prüfereignisse	REST	Nein	Ja	Ja	Ja	Nein
Kampagnen	REST	Nein	Ja	Ja	Ja	Nein
Interaktionen	REST	Nein	Ja	Ja	Ja	Nein

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt SELECT *	Unterstützt Partitionierung
Inhalte und Ressourcen	REST	Nein	Ja	Ja	Ja	Nein

Beispiel für REST:

```
salesforcemarketingcloud _read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Campaigns",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://**********.rest.marketingcloudapis.com"
    }
```

Beispiel für SOAP:

```
salesforcemarketingcloud _read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Activity",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://**********.soap.marketingcloudapis.com"
    }
```

Angaben zur Salesforce Marketing Cloud-Entität und zum Feld:

In den folgenden Tabellen werden die Salesforce Marketing Cloud-Entitäten beschrieben. Es gibt REST-Entitäten mit statischen Metadaten und SOAP-Entitäten mit dynamischen Metadaten.

REST-Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren
Rückruf bei Ereignisbenachrichtigung	Rückruf-ID	String	
	Name des Rückrufs	String	
	URL	String	
	maxBatchSize	Ganzzahl	
	Status	String	
	Grund für den Status	String	
Samenliste	id	String	
	Name	String	
	description	String	
	activeSeedCount	Ganzzahl	
Aufstellen	Kundenschlüssel	String	
	Name	String	
	description	String	
	Art des Standorts	String	'='
	awsFileTransferSpeicherort	Struct	
Überprüfung der Domain	Unternehmens-ID	Ganzzahl	
	Status	String	'='
	Typ der Domäne	String	'='
	Mitglieds-ID	Ganzzahl	

Entität	Feld	Datentyp	Unterstützte Operatoren
	emailSendTime	DateTime	
	Domain	String	
	Sendbar	Boolesch	
Objekte, verschachtelte Tags	id	Ganzzahl	
	Änderungsdatum	DateTime	
	tags	Auflisten	
	Name	String	
	description	String	
	ID des Elternteils	Ganzzahl	
Kontakt	values	Auflisten	
Abonnement für Ereignisbenachrichtigungen	Name des Abonnements	String	
	Rückruf-ID	String	
	Name des Rückrufs	String	
	eventCategoryTypes	Auflisten	
	-Filter	Auflisten	
	URL	String	
	maxBatchSize	Ganzzahl	
	Abonnement-ID	String	
	Status	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Grund für den Status	String	
Messaging	Zeit der Lieferung	DateTime	
	id	String	
	messageId	String	
	Status	String	
	to	Struct	
Interaktionen	Status	String	'='
	id	String	
	Schlüssel	String	
	Name	String	
	lastPublishedDate	DateTime	
	description	String	
	version	Ganzzahl	
	workflowApiVersion	Ganzzahl	
	Erstellungsdatum	DateTime	
	Datum geändert	DateTime	
	Ziele	Struct	
	stats	Struct	
	Eingabemodus	String	
	defaults	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Ausführungsmodus	Struct	
	Definitions-ID	String	
Inhalte und Ressourcen	id	Ganzzahl	
	Kundenschlüssel	String	
	objectId	String	
	contentType	String	
	Art der Anlage	Struct	
	Name	String	
	description	String	
	owner	Struct	
	Erstellungsdatum	DateTime	
	Erstellt von	Struct	
	Datum geändert	DateTime	
	Geändert von	Struct	
	Miniaturansicht	Struct	
	Kategorie	Struct	
	Meta	Struct	
	Ansichten	Struct	
	Verfügbare Ansichten	Struct	
	data	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Legacy-Daten	Struct	
	Modellversion	Ganzzahl	
	Version	Ganzzahl	
	Gesperrt	Boolesch	
	FileProperties	Struct	
	Tags	Auflisten	
	Inhalt	String	
	Design	String	
	SuperContent	String	
	CustomFields	Struct	
	Blöcke	Struct	
	MinBlocks	Ganzzahl	
	MaxBlocks	Ganzzahl	
	Kanäle	Struct	
	AllowedBlocks	Auflisten	
	Slots	Struct	
	BusinessUnitAvailability	Struct	
	Eigenschaften teilen	Struct	
	Eigenschaften teilen. Geteilt mit	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	SharingProperties. SharingType	String	
	Vorlage	Struct	
	Datei	String	
	GenerateFrom	String	
Ereignisse prüfen	id	Ganzzahl	
	Erstellungsdatum	DateTime	
	Mitglieds-ID	Ganzzahl	
	Unternehmens-ID	Ganzzahl	
	Mitarbeiter	Struct	
	objectType	Struct	
	operation	Struct	
	object	Struct	
	Transaktions-ID	String	
Kampagnen	id	Ganzzahl	
	Erstellungsdatum	DateTime	
	Datum geändert	DateTime	
	Name	String	
	description	String	
	Kampagnencode	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	color	String	
	Favorit	Boolesch	

SOAP-Entitäten mit dynamischen Metadaten:

Entität	Datentyp	Unterstützte Operatoren
Aktivität	String	WIE,! =, =
	Struct	
	Ganzzahl	!=,=,>=,<=,<,>
	Double	!=,=,>=,<=,<,>
	Boolesch	!=,=
	DateTime	>=, <=, <, >, =, ZWISCHEN
Ereignis „Bounce“	Ganzzahl	!=,=,>=,<=,<,>
	DateTime	>=, <=, <, >, =, ZWISCHEN
	String	WIE,! =, =
	Struct	
Klicken Sie auf Event	Ganzzahl	!=,=,>=,<=,<,>
	DateTime	>=, <=, <, >, =, ZWISCHEN
	String	WIE,! =, =
	Struct	
Inhaltsbereich	Struct	

Entität	Datentyp	Unterstützte Operatoren
	String	WIE,! =, =
	Ganzzahl	!=,=,>=,<=,<,>
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Boolesch	!=,=
Erweiterung der Daten	DateTime	>=, <=, <, >, =, ZWISCHEN
Email	String	WIE,! =, =
	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Boolesch	!=,=
	Struct	
Ereignis „Weitergeleitete E-Mail“	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Weitergeleitete E-Mail OptInEvent	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Link	Ganzzahl	!=,=,>=,<=,<,>

Entität	Datentyp	Unterstützte Operatoren
Link Senden	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	Double	!=,=,>=,<=,<,>
Auflisten	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Abonnent auflisten	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Ereignis nicht gesendet	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Veranstaltung öffnen	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Senden	Ganzzahl	!=,=,>=,<=,<,>

Entität	Datentyp	Unterstützte Operatoren
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Boolesch	!=,=
	Struct	
Gesendetes Ereignis	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Subscriber	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Veranstaltung zur Umfrage	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Struct	
Unsub-Ereignis	Ganzzahl	!=,=,>=,<=,<,>
	String	WIE,! =, =
	DateTime	>=, <=, <, >, =, ZWISCHEN
	Boolesch	!=,=

Entität	Datentyp	Unterstützte Operatoren
	Struct	

Abfragen partitionieren

In Salesforce Marketing Cloud unterstützen die Felder Integer und DateTime Datatype die feldbasierte Partitionierung.

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Zeitstempelfeld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

  ```
  “2024-05-07T02:03:00.00Z"
  ```
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
salesforcemarketingcloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "ListSubscriber",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "CreatedDate"
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z"
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z"
```

```
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für die Salesforce Marketing Cloud

Im Folgenden sind die Verbindungsoptionen für Salesforce Marketing Cloud aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Salesforce Marketing Cloud.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Rest- und SOAP-API-Version von Salesforce Marketing Cloud, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.
- `WRITE_OPERATION`(Zeichenfolge) — Standard: INSERT. Wird zum Schreiben verwendet. Der Wert sollte INSERT, UPDATE, UPSERT sein.
- `ID_FIELD_NAMES`(Zeichenfolge) — Standard: null. Erforderlich für UPDATE/UPSERT.

Einschränkungen und Hinweise für den Salesforce Marketing Cloud Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Salesforce Marketing Cloud-Connector:

- Wenn Sie den Filter für ein DateTime Datentypfeld verwenden, müssen Sie den Wert im Format "yyyy-mm-ddthh:mm:ssz“ übergeben.

- In der Datenvorschau wird der boolesche Datentypwert leer angezeigt.
- Für SOAP-Entitäten können Sie maximal zwei Filter definieren, und für REST-Entitäten können Sie nur einen Filter definieren, wodurch das Testen der Partitionierung mit Filtern eingeschränkt wird.
- Auf SaaS-Seite wurden mehrere unerwartete Verhaltensweisen beobachtet: Das `Link.Alias` Feld in der Entität `linksend` unterstützt den Operator CONTAINS nicht (zum Beispiel`Link.Alias CONTAINS "ViewPrivacyPolicy"`), und Filteroperatoren für Data Extension-Entitäten (wie EQUALS und GREATER THAN) geben nicht die erwarteten Ergebnisse zurück.
- Die ClickEvent SFMC-SOAP-API hat eine Verzögerung bei der Wiedergabe neu erstellter Datensätze, sodass die kürzlich erstellten Datensätze möglicherweise nicht sofort in der API-Antwort verfügbar sind.

  Beispiel: Wenn Sie am 2025-01-10T 14:30:00 5 neue ClickEvent Datensätze erstellen und diese sofort über die SOAP-API abrufen, enthält die Antwort möglicherweise nicht alle 5 Datensätze. Es kann bis zu 5 Minuten dauern, bis die neu erstellten Datensätze in der API-Antwort erscheinen. Diese Verzögerung kann sich sowohl auf den Datenabruf als auch auf geplante Datenläufe auswirken.
- Zwei verschiedene DateTime Formate: 2025-03-11T 04:46:00 (ohne Millisekunden) und 2025-03-11T 04:46:00.000 Z werden unterstützt, wenn Schreibvorgänge in (mit Millisekunden) ausgeführt werden. AWS Glue
- Für die Entität Event Notification Subscription kann ein Abonnement nur für eine verifizierte Rückruf-URL erstellt werden, und Sie können bis zu 200 Abonnements pro Rückruf haben.
- Für die Entität Event Notification Callback können maximal 50 Datensätze pro Konto erstellt werden.

Verbindung zur Salesforce Commerce Cloud herstellen

Die B2C-Commerce-API ist eine Sammlung von RESTful APIs für die Interaktion mit B2C-Commerce-Instanzen. Sie hat verschiedene Namen: Salesforce Commerce API, das Akronym SCAPI oder einfach Commerce API.

Die API ermöglicht es Entwicklern, eine breite Palette von Anwendungen zu erstellen: von vollständigen Storefronts bis hin zu benutzerdefinierten Händlertools zur Erweiterung von Business Manager. Für alle B2C-Commerce-Kunden ist die API ohne zusätzliche Kosten verfügbar.

Die API ist in zwei Hauptgruppen unterteilt APIs: Shopper APIs und Admin. APIs Und innerhalb jeder Gruppe sind sie in API-Familien und in kleinere Gruppen unterteilt, die sich auf verwandte Funktionen konzentrieren.

Themen

- AWS Glue Unterstützung für Salesforce Commerce Cloud
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration der Salesforce Commerce Cloud
- Konfiguration von Salesforce Commerce Cloud-Verbindungen
- Aus Salesforce Commerce Cloud-Entitäten lesen
- Referenz zu den Salesforce Commerce Cloud-Verbindungsoptionen
- Einschränkungen

AWS Glue Unterstützung für Salesforce Commerce Cloud

AWS Glue unterstützt Salesforce Commerce Cloud wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus der Salesforce Commerce Cloud abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Salesforce Commerce Cloud-API-Versionen

v1.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration der Salesforce Commerce Cloud

Bevor Sie Daten aus der Salesforce Commerce Cloud übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

- Sie haben eine Salesforce Commerce Cloud-Kundenanwendung mit clientId und ClientSecret.

- Ihr Salesforce Commerce Cloud-Konto ist für den API-Zugriff aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce Commerce Cloud-Konto herstellen. Für typische Verbindungen müssen Sie in der Salesforce Commerce Cloud nichts weiter tun.

Konfiguration von Salesforce Commerce Cloud-Verbindungen

Salesforce Commerce Cloud unterstützt den Gewährungstyp CLIENT CREDENTIALS für OAuth2.

- Dieser Gewährungstyp gilt als 2-beinig OAuth 2.0, da er von Kunden verwendet wird, um außerhalb des Benutzerkontextes ein Zugriffstoken zu erhalten. AWS Glue ist in der Lage, die Client-ID und das Client-Geheimnis zur Authentifizierung von Salesforce Commerce Cloud zu verwenden APIs , die von benutzerdefinierten Diensten bereitgestellt werden, die Sie definieren.
- Jeder benutzerdefinierte Service gehört einem Nur-API-Benutzer, der über eine Reihe von Rollen und Berechtigungen verfügt, die den Service autorisieren, bestimmte Aktionen auszuführen. Ein Zugriffstoken ist einem einzelnen benutzerdefinierten Dienst zugeordnet.
- Dieser Gewährungstyp führt zu einem Zugriffstoken, das nur von kurzer Dauer ist und durch Aufrufen des Identitätsendpunkts erneuert werden kann.
- Weitere Informationen zur Salesforce Commerce Cloud-Dokumentation zur Generierung der Kundenanmeldedaten finden Sie in der Salesforce-Dokumentation.

So konfigurieren Sie eine Salesforce Commerce Cloud-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in einen geheimen Schlüssel zu erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps sollte Secret das Consumer Secret der verbundenen App mit USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET als Schlüssel enthalten.
2. Stellen Sie unter Datenverbindungen eine AWS Glue Studio Verbindung her, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie unter Datenverbindungen die Option Verbindung erstellen aus.
   b. Wählen Sie bei der Auswahl einer Datenquelle Salesforce Commerce Cloud aus.
   c. Geben Sie Ihren Salesforce Commerce Cloud-Kurzcode, Ihre Organisations-ID und Ihre Site-ID an.

d. Wählen Sie die Salesforce Commerce Cloud-Domain-URL Ihres Salesforce Commerce Cloud-Kontos aus.

e. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

f. Geben Sie die OAuth Bereiche an — optional, die benutzerverwaltete Client-Anwendung ClientId der Salesforce Commerce Cloud, zu der Sie eine Verbindung herstellen möchten.

g. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

h. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
4. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Salesforce Commerce Cloud-Entitäten lesen

Voraussetzungen

- Ein Salesforce Commerce Cloud-Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Zuweisungen	Ja	Ja	Ja	Ja	Ja
Kampagnen	Ja	Ja	Ja	Ja	Ja
Kataloge	Ja	Ja	Ja	Ja	Ja
Kategorien	Ja	Ja	Ja	Ja	Ja
Gutscheine	Ja	Ja	Ja	Ja	Ja
Geschenkgutscheine	Ja	Ja	Ja	Ja	Ja
Produkte	Ja	Ja	Ja	Ja	Ja
Werbeaktionen	Ja	Ja	Ja	Ja	Ja
Quellcode-Gruppen	Ja	Ja	Ja	Ja	Ja

Beispiel

```
salesforce_commerce_cloud_read = glueContext.create_dynamic_frame.from_options(
     connection_type="SalesforceCommerceCloud",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "campaign",
         "API_VERSION": "v1"
     }
```

Angaben zur Salesforce Commerce Cloud-Entität und zum Feld

Liste der Entitäten:

- Aufgaben: https://developer.salesforce.com/docs/commerce/commerce-api/references/assignments
- Kampagnen: https://developer.salesforce.com/docs/commerce/commerce-api/references/campaigns
- Kataloge: https://developer.salesforce.com/docs/commerce/commerce-api/references/catalogs
- Kategorien:? https://developer.salesforce.com/docs/ commerce/commerce-api/references/catalogs meta=Kategorien suchen
- Geschenkgutscheine: -Zertifikate https://developer.salesforce.com/docs/ commerce/commerce-api/references/gift
- Produkte: https://developer.salesforce.com/docs/commerce/commerce-api/references/products
- Werbeaktionen: https://developer.salesforce.com/docs/commerce/commerce-api/references/promotions
- Quellcode-Gruppen: https://developer.salesforce.com/docs/commerce/commerce-api/references/source-code-groups

Abfragen partitionieren

Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für gültige Werte:`"2024-02-06"`.
- `UPPER_BOUND`: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt:

Entität	Feld partitionieren	DataType
Kampagnen	Zuletzt geändert	DateTime
Kampagnen	startDate	DateTime
Kampagnen	endDate	DateTime
Kataloge	creationDate	DateTime
Kategorien	Erstellungsdatum	DateTime
Geschenkgutscheine	Händler-ID	String
Geschenkgutscheine	Erstellungsdatum	DateTime
Produkte	Erstellungsdatum	DateTime
Produkte	Letzte Änderung	DateTime
Quellcode-Gruppen	creationDate	DateTime
Quellcode-Gruppen	startTime	DateTime
Quellcode-Gruppen	endTime	DateTime

Beispiel

```
salesforceCommerceCloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforceCommerceCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "creationDate"
        "LOWER_BOUND": "2020-05-01T20:55:02.000Z"
        "UPPER_BOUND": "2024-07-11T20:55:02.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Referenz zu den Salesforce Commerce Cloud-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Salesforce Commerce Cloud aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Salesforce Commerce Cloud.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Die Salesforce Commerce Cloud-Rest-API-Version, die Sie verwenden möchten. Beispiel: v1.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Salesforce Commerce Cloud-Connector:

- Der Filter Enthält funktioniert bei der Partitionierung nicht wie erwartet.
- Die Entität von CDN Zones unterstützt keine Sandbox-Instanzen und unterstützt nur Entwicklungs- und Produktionsinstanztypen. Weitere Informationen finden Sie unter ArticleView? https:// help.salesforce.com/s/ id=cc.b2c_embedded_cdn_overview.htm.
- In Salesforce Commerce Cloud gibt es keinen API-Endpunkt zum Abrufen dynamischer Metadaten. Daher gibt es keine Möglichkeit, die benutzerdefinierten Felder in der Produkt- und Kategorieentität zu unterstützen.

- Die Site-ID ist ein obligatorischer Abfrageparameter. Sie müssen den Wert für die Site-ID über die benutzerdefinierte Connector-Einstellung übergeben. Weitere Informationen finden Sie unter Basis-URL und Anforderungsformierung.
- Sie können Filter auf maximal zwei Felder (ohne Ebenen, falls vorhanden) in einer einzigen API-Anfrage mit der Kombination verschiedener Operatoren anwenden, wie in der folgenden Tabelle beschrieben:

Filterkriterien	Wird unterstützt?
Ein Feld mit dem Operator CONTAINS in einer einzigen API-Anfrage.	Ja
Ein Feld mit dem Equals-Operator in einer einzigen API-Anfrage.	Ja
Ein Feld mit dem Operator BETWEEN in einer einzelnen API-Anfrage.	Ja
Zwei oder mehr Felder mit dem Operator CONTAINS in einer einzigen API-Anfrage.	Nein
Zwei oder mehr Felder mit dem Equals-Operator in einer einzigen API-Anfrage.	Nein
Zwei oder mehr Felder mit dem Operator BETWEEN in einer einzigen API-Anfrage.	Nein
Ein Feld mit Equals und ein Feld mit dem Operator CONTAINS in einer einzigen API-Anfrage.	Ja
Ein Feld mit BETWEEN und ein Feld mit dem CONTAINS-Operator in einer einzigen API-Anfrage.	Ja
Ein Feld mit BETWEEN und ein Feld mit dem Equals-Operator in einer einzigen API-Anfrage.	Ja

Filterkriterien	Wird unterstützt?
Ein Feld mit Equals, ein Feld mit CONTAINS und ein Feld mit dem BETWEEN-Operator in einer einzigen API-Anfrage.	Nein
Ein Feld mit dem Equals-Operator, wenn INCREMENTAL PULL in einer einzelnen API-Anfrage angewendet wird.	Ja
Ein Feld mit dem Operator CONTAINS, wenn INCREMENTAL PULL in einer einzelnen API-Anfrage angewendet wird.	Ja
Ein Feld mit dem Operator BETWEEN, wenn INCREMENTAL PULL in einer einzelnen API-Anfrage angewendet wird.	Nein
Ein Equals- und ein CONTAINS-Operator, wenn INCREMENTAL PULL in einer einzelnen API-Anfrage angewendet wird.	Nein

- In einigen Entitäten unterscheidet sich der Datentyp für die Felder beim Abrufen von Feldern von dem, wenn sie als durchsuchbare Felder verwendet werden. Aus diesem Grund gibt es keine Filterfunktion für diese Felder. Die folgende Tabelle enthält Einzelheiten zu solchen Feldern.

Herr Nr.	Entity Name	Durchsuchbarer Feldname	Datentyp als durchsuchbar	Datentyp als abrufbar
1	Katalog	Name	String	Struct
2	Katalog	description	String	Struct
3	Kategorie	Name	String	Struct
4	Kategorie	description	String	Struct
5	Produkt	Name	String	Struct

Herr Nr.	Entity Name	Durchsuchbarer Feldname	Datentyp als durchsuchbar	Datentyp als abrufbar
6	Produkt	durchsuchbar	Boolesch	Struct
7	Produkt	Gültig von	DateTime	Struct
8	Produkt	Gültig bis	DateTime	Struct
9	Produkt	Typ	String	Struct
10	Produkt	Online-Flagge	Boolesch	Struct
11	Beförderung	Name	String	Struct

Verbindung mit Salesforce Marketing Cloud Account Engagement herstellen

Salesforce Marketing Cloud Account Engagement ist eine Marketing-Automatisierungslösung, die Unternehmen dabei unterstützt, sinnvolle Verbindungen aufzubauen, mehr Pipeline zu generieren und den Vertrieb in die Lage zu versetzen, mehr Geschäfte abzuschließen. Wenn Sie ein Salesforce Marketing Cloud Account Engagement-Benutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce Marketing Cloud Account Engagement-Konto herstellen. Sie können Salesforce Marketing Cloud Account Engagement als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten von Salesforce Marketing Cloud Account Engagement an AWS Dienste oder andere unterstützte Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Salesforce Marketing Cloud Account Engagement
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration der Salesforce Marketing Cloud-Kontobindung
- Konfiguration von Salesforce Marketing Cloud-Account-Engagement-Verbindungen
- Lesen aus Salesforce Marketing Cloud Account Engagement-Entitäten
- Verbindungsoptionen für Salesforce Marketing Cloud Account Engagement
- Einschränkungen und Hinweise für den Salesforce Marketing Cloud Account Engagement Connector

AWS Glue Unterstützung für Salesforce Marketing Cloud Account Engagement

AWS Glue unterstützt Salesforce Marketing Cloud Account Engagement wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Salesforce Marketing Cloud Account Engagement entweder im asynchronen Modus oder im Synchronisierungsmodus abzufragen.

Als Ziel unterstützt?

Nein.

Unterstützte Versionen der Salesforce Marketing Cloud Account Engagement API

Die folgenden Versionen der Salesforce Marketing Cloud Account Engagement API werden unterstützt:

- v5

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration der Salesforce Marketing Cloud-Kontobindung

Bevor Sie AWS Glue Daten aus Salesforce Marketing Cloud Account Engagement übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Salesforce-Marketingkonto.
- Sie haben einen lizenzierten Account Engagement-Plan für das Salesforce-Konto.
- Sie haben den Salesforce-Benutzer mit dem Account Engagement-Benutzer synchronisiert.
- Sie haben unter App Manager eine neue verbundene Anwendung erstellt, um OAuth Anmeldeinformationen zu erhalten.

Wenn Sie diese Anforderungen erfüllen, sind Sie bereit, eine Verbindung AWS Glue zu Ihrem Salesforce Marketing Cloud Account Engagement-Konto herzustellen.

Konfiguration von Salesforce Marketing Cloud-Account-Engagement-Verbindungen

Die Art der Gewährung bestimmt, wie mit Salesforce Marketing Cloud Account Engagement AWS Glue kommuniziert wird, um Zugriff auf Ihre Daten anzufordern. Ihre Wahl wirkt sich auf die Anforderungen aus, die Sie erfüllen müssen, bevor Sie die Verbindung herstellen. Salesforce Marketing Cloud Account Engagement unterstützt nur den Zuschusstyp AUTHORIZATION_CODE für 2.0. OAuth

- Dieser Gewährungstyp gilt als „dreibeiniges“, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene Anwendung in Salesforce Marketing Cloud Account Engagement zu erstellen und beim Herstellen von Verbindungen über die AWS Glue Konsole ihre eigene Kunden-ID und ihren eigenen geheimen Kundenschlüssel anzugeben. In diesem Szenario werden sie weiterhin zu Salesforce Marketing Cloud Account Engagement weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
- Die öffentliche Dokumentation zu Salesforce Marketing Cloud Account Engagement zur Erstellung einer verbundenen Anwendung für den OAuth Autorisierungscode-Flow finden Sie unter Authentifizierung.

So konfigurieren Sie eine Salesforce Marketing Cloud-Account-Engagement-Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Salesforce Marketing Cloud Account Engagement aus.
   b. Geben Sie die `INSTANCE_URL` Salesforce Marketing Cloud Account Engagement-Instanz an, mit der Sie eine Verbindung herstellen möchten.
   c. Geben Sie die `PARDOT_BUSINESS_UNIT_ID` Salesforce Marketing Cloud Account Engagement-Instanz an, mit der Sie eine Verbindung herstellen möchten.
   d. Wählen Sie die entsprechende Autorisierungscode-URL aus der Dropdownliste aus.

e. Wählen Sie die entsprechende Token-URL aus der Dropdownliste aus.

f. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

g. Geben Sie die Client-ID der vom Benutzer verwalteten Client-Anwendung an (die Client-ID der verbundenen App).

h. Wählen Sie `secretName` die aus, in AWS Glue die Sie für diese Verbindung die Token einfügen möchten. Das ausgewählte Geheimnis muss über einen Schlüssel `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` verfügen, dessen Wert dem geheimen Clientschlüssel der verbundenen App entspricht.

i. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
3. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen aus Salesforce Marketing Cloud Account Engagement-Entitäten

Voraussetzung

Ein Salesforce Marketing Cloud-Objekt zur Kundenbindung, aus dem Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für Sync-Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kampagne	Ja	Ja	Ja	Ja	Ja
Dynamischer Inhalt	Ja	Ja	Ja	Ja	Ja
Email	Ja	Ja	Ja	Ja	Ja
E-Mail-Vorlage	Ja	Ja	Ja	Ja	Ja
Engagement Studio-Programm	Ja	Ja	Ja	Ja	Ja
Inhalt des Ordners	Ja	Ja	Ja	Ja	Ja
Landing-Page	Ja	Ja	Ja	Ja	Ja
Verlauf des Lebenszyklus	Ja	Ja	Ja	Ja	Ja
Lebenszyklus-Phase	Ja	Ja	Ja	Ja	Ja
Auflisten	Ja	Ja	Ja	Ja	Ja
E-Mail auflisten	Ja	Ja	Ja	Ja	Ja
Mitgliedschaft auflisten	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Gelegenheit	Ja	Ja	Ja	Ja	Ja
Aussicht	Ja	Ja	Ja	Ja	Ja
Konto für potenzielle Kunden	Ja	Ja	Ja	Ja	Ja
Benutzer	Ja	Ja	Ja	Ja	Ja

Beispiel:

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforcePardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5"
    }
   )
```

Unterstützte Entitäten für die Async-Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Kampagne	Ja	Nein	Nein	Ja	Nein
Dynamischer Inhalt	Ja	Nein	Nein	Ja	Nein
E-Mail-Vorlage	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Landingpage	Ja	Nein	Nein	Ja	Nein
Verlauf des Lebenszyklus	Ja	Nein	Nein	Ja	Nein
Lebenszyklus-Phase	Ja	Nein	Nein	Ja	Nein
Auflisten	Ja	Nein	Nein	Ja	Nein
E-Mail auflisten	Ja	Nein	Nein	Ja	Nein
Mitgliedschaft auflisten	Ja	Nein	Nein	Ja	Nein
Gelegenheit	Ja	Nein	Nein	Ja	Nein
Aussicht	Ja	Nein	Nein	Ja	Nein
Konto für potenzielle Kunden	Ja	Nein	Nein	Ja	Nein
Benutzer	Ja	Nein	Nein	Ja	Nein

Beispiel:

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforcePardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5",
        "TRANSFER_MODE": "ASYNC"
    }
```

```
)
```

Angaben zur Entität und zum Feld Salesforce Marketing Cloud Account Engagement:

Um die Felddetails für die folgenden Entitäten anzuzeigen, navigieren Sie zur Salesforce Marketing Cloud Account Engagement API, wählen Sie Guides aus, scrollen Sie nach unten zu Open Source API Wrappers, erweitern Sie Version 5-Dokumente im Menü und wählen Sie eine Entität aus.

Liste der Entitäten:

- Kampagne
- Dynamischer Inhalt
- Email
- E-Mail-Vorlage
- Engagement Studio-Programm
- Inhalt des Ordners
- Landing-Page
- Verlauf des Lebenszyklus
- Lebenszyklus-Phase
- Auflisten
- E-Mail auflisten
- Mitgliedschaft auflisten
- Gelegenheit
- Aussicht
- Konto für potenzielle Kunden
- Benutzer

Zusätzlich zu den oben genannten Feldern unterstützt der asynchrone Modus spezifische filterbare Felder für jede Entität, wie in der folgenden Tabelle dargestellt.

Entität	Zusätzliche filterbare Felder werden in Async unterstützt
Kampagne	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Dynamischer Inhalt	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
E-Mail-Vorlage	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Engagement Studio-Programm	-
Landingpage	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Verlauf des Lebenszyklus	`createdAfter` , `createdBefore`
Lebenszyklus-Phase	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Auflisten	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
E-Mail auflisten	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Mitgliedschaft auflisten	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`

Entität	Zusätzliche filterbare Felder werden in Async unterstützt
Gelegenheit	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Aussicht	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Konto für potenzielle Kunden	`createdAfter` , `createdBefore` , `deleted`
Benutzer	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`

Weitere Informationen zu den zusätzlichen Feldern finden Sie unter Salesforce Export API

Beachten Sie die folgenden Überlegungen zum Konnektor:

- Der Wert des `delete` Felds in den Entitäten kann `false` (Standard)`true`, oder sein`all`.

Partitionierung von Abfragen

Filterbasierte Partitionierung:

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,, und angeben `LOWER_BOUND``UPPER_BOUND`, `NUM_PARTITIONS` ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in SPark SQL-Abfragen verwendet wird.

Beispiele für gültige Werte:

```
"2022-01-01T01:01:01.000Z"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.
- PARTITION_BY: die Art der durchzuführenden Partitionierung. Bei feldbasierter Partitionierung ist „FIELD“ zu übergeben.

Beispiel:

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcepardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2022-01-01T01:01:01.000Z"
        "UPPER_BOUND": "2024-01-01T01:01:01.000Z"
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "FIELD"
    }
   )
```

Verbindungsoptionen für Salesforce Marketing Cloud Account Engagement

Im Folgenden sind die Verbindungsoptionen für Salesforce Marketing Cloud Account Engagement aufgeführt:

- ENTITY_NAME(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Salesforce Marketing Cloud Account Engagement.
- PARDOT_BUSINESS_UNIT_ID— (Erforderlich) Wird zum Herstellen einer Verbindung verwendet. Die Geschäftsbereichs-ID der Salesforce Marketing Cloud Account Engagement-Instanz, mit der Sie eine Verbindung herstellen möchten.

- API_VERSION(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die restliche API-Version der Salesforce Marketing Cloud Account Engagement, die Sie verwenden möchten.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) -
  - Im Synchronisierungsmodus — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
  - Im asynchronen Modus — Standard: Aktueller DateTime Wert (gemäß der Zeitzone des Benutzers) — 1 Jahr. Wird zum Lesen verwendet.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- UPPER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- INSTANCE_URL(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige URL für die Salesforce Marketing Cloud-Account-Engagement-Instanz.
- PARTITION_BY(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Art der durchzuführenden Partitionierung. Bei feldbasierter Partitionierung ist „FIELD“ zu übergeben.
- TRANSFER_MODE(String) — (Optional), Wert, der für die Ausführung eines Jobs im ASYNC-Modus verwendet werden soll. Wenn diese Option nicht angegeben wird, wird der Job im SYNC-Modus ausgeführt.

Einschränkungen und Hinweise für den Salesforce Marketing Cloud Account Engagement Connector

Es gelten die folgenden Hinweise und Einschränkungen:

- Wenn sowohl ein Limit als auch eine Partitionierung angewendet werden, hat das Limit Vorrang vor der Partitionierung.

- Gemäß den API-Dokumenten wurde bei täglichen und `SalesforceMarketingCloudEngagement` gleichzeitigen Anfragen ein RateLimit erzwungen. Weitere Informationen finden Sie unter Ratenlimits.
- Die Export-API unterliegt dem täglichen Account-Engagement-API-Aufruflimit und dem Limit für gleichzeitige Account-Engagement-API-Aufrufe für Ihr Konto.
- Ähnlich wie bei einer Warteschlange werden Export/Async-API-Aufrufe nacheinander für jedes Konto ausgeführt. Ältere Exporte werden vor neueren Exporten verarbeitet.
- Die Partition wird im asynchronen Modus nicht unterstützt.
- Die Anzahl der ausgewählten Felder, die in den Export/Async-API-Aufrufen angegeben wurden, darf 150 nicht überschreiten.
- Die Prospect-Entität unterstützt über 150 Felder, es können jedoch nur 150 Felder gleichzeitig ausgewählt werden. Wenn diese Option ausgewählt `Select All` ist, werden einige Felder ausgeschlossen. Um Daten für diese ausgeschlossenen Felder abzurufen, müssen Sie sie in die `Selected Fields` Option aufnehmen.

  Die folgende Liste enthält die ausgeschlossenen Felder in `SELECT_ALL` - `updatedBy.firstNameupdatedBy.lastName`,`updatedBy.jobTitle`, `updatedBy.roleNameupdatedBy.salesforceId`,`updatedBy.createdAt`,`updatedBy.updatedAt` `updatedBy.tagReplacementLanguage`
- Sammlungsfelder können nicht für Async exportiert werden. Beispielsweise werden in List-E-Mail die `replyToOptions` Felder `senderOptions` und nicht unterstützt.
- Für alle Entitäten ist der Filter obligatorisch. Wenn kein Filter angegeben wird, wird das Standardfilterprädikat auf das `Created After` Feld mit dem Wert des aktuellen Datums und der Uhrzeit (angepasst an Ihre Zeitzone) minus einem Jahr gesetzt.
- Gemäß den Beschränkungen für die Kundenbindung in Salesforce Marketing Cloud beträgt der maximale Bereich für das Abrufen von Daten in Async 1 Jahr. Wenn eine Abfrage länger als ein Jahr bereitgestellt wird, gibt der Job einen Fehler aus.
- Derzeit gibt es einen Fehler in Salesforce Pardot. Wenn der Job nur ein einzelnes Feld enthält, das keine Daten enthält, gibt der Feldwert kein korrektes Ergebnis zurück, sondern der Feldname wird mehrmals zurückgegeben. Das Salesforce Pardot-Team ist sich des Problems bewusst und arbeitet aktiv an einer Lösung.

Verbindung zu SAP HANA herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für SAP HANA. AWS Glue Studio bietet eine visuelle Oberfläche, über die Sie eine Verbindung zu SAP HANA herstellen, Datenintegrationsjobs erstellen und diese auf dem ausführen können AWS Glue Studio serverlose Spark-Laufzeit.

AWS Glue Studio erstellt eine einheitliche Verbindung für SAP HANA. Weitere Informationen finden Sie unter Überlegungen.

Themen

- Erstellen einer SAP-HANA-Verbindung
- Erstellen eines SAP-HANA-Quellknotens
- Erstellen eines SAP-HANA-Zielknotens
- Erweiterte Optionen

Erstellen einer SAP-HANA-Verbindung

Um von aus eine Verbindung zu SAP HANA herzustellen AWS Glue, müssen Sie Ihre SAP HANA-Anmeldeinformationen erstellen und in einem AWS Secrets Manager geheimen Ordner speichern und dieses Geheimnis dann einer SAP AWS Glue HANA-Verbindung zuordnen. Sie müssen die Netzwerkkonnektivität zwischen Ihrem SAP-HANA-Service und AWS Glue konfigurieren.

Voraussetzungen:

- Wenn sich Ihr SAP HANA-Service in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue Job mit dem SAP HANA-Service kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die bei der Ausführung des Jobs verwendet AWS Glue werden. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrem SAP-HANA-Endpunkt und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem SAP-HANA-JDBC-Port herstellen. Weitere Informationen zu SAP-HANA-Ports finden Sie in der Dokumentation von SAP HANA. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

Konfigurieren Sie eine Verbindung zu SAP HANA wie folgt:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer SAP HANA-Anmeldeinformationen ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *saphanaUsername*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *saphanaPassword*

2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen,*connectionName*, für die future Verwendung in AWS Glue.

   - Wählen Sie als Verbindungstyp SAP HANA aus.
   - Wenn Sie die SAP-HANA-URL angeben, geben Sie die URL für Ihre Instance an.

     SAP HANA JDBC URLs haben das folgende Format `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWS Glue erfordert die folgenden JDBC-URL-Parameter:

     - `databaseName` – Eine Standarddatenbank in SAP HANA, mit der eine Verbindung hergestellt werden kann.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an. *secretName*

Nachdem Sie eine AWS Glue SAP HANA-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue Job ausführen können:

- Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle *secretName* Leserechte.

Erstellen eines SAP-HANA-Quellknotens

Voraussetzungen

- Eine AWS Glue SAP HANA-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer SAP-HANA-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine SAP-HANA-Tabelle, aus der Sie lesen*`tableName`*, oder abfragen möchten*`targetQuery`*.

  Eine Tabelle kann mit einem SAP-HANA-Tabellennamen und -Schemanamen im folgenden Format angegeben werden: *`schemaName`*`.`*`tableName`*. Der Schemaname und das Trennzeichen „.“ sind nicht erforderlich, wenn sich die Tabelle im Standardschema „public“ befindet. Nennen Sie das*`tableIdentifier`*. Beachten Sie, dass die Datenbank als JDBC-URL-Parameter in `connectionName` bereitgestellt wird.

Hinzufügen einer SAP-HANA-Datenquelle

Einen Knoten Datenquelle – SAP HANA fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre SAP-HANA-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie SAP-HANA-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer SAP-HANA-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie eine Option für SAP-HANA-Quelle aus:

   - Eine einzelne Tabelle auswählen – rufen Sie alle Daten aus einer einzigen Tabelle ab.
   - Benutzerdefinierte Abfrage eingeben – rufen Sie einen Datensatz aus mehreren Tabellen ab, basierend auf Ihrer benutzerdefinierten Abfrage.
3. Wenn Sie eine einzelne Tabelle ausgewählt haben, geben Sie ein*`tableName`*.

   Wenn Sie Benutzerdefinierte Abfrage eingeben ausgewählt haben, geben Sie eine SQL-SELECT-Abfrage ein.

4. Geben Sie unter Benutzerdefinierte SAP-HANA-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines SAP-HANA-Zielknotens

Voraussetzungen

- Eine AWS Glue SAP HANA-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer SAP-HANA-Verbindung”.
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine SAP-HANA-Tabelle, in die Sie schreiben möchten,*tableName*.

  Eine Tabelle kann mit einem SAP-HANA-Tabellennamen und -Schemanamen im folgenden Format angegeben werden: *schemaName*.*tableName*. Der Schemaname und das Trennzeichen „.“ sind nicht erforderlich, wenn sich die Tabelle im Standardschema „public“ befindet. Nennen Sie das*tableIdentifier*. Beachten Sie, dass die Datenbank als JDBC-URL-Parameter in `connectionName` bereitgestellt wird.

Hinzufügen eines SAP-HANA-Datenziels

Einen Knoten Datenziel – SAP HANA fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre SAP-HANA-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie SAP-HANA-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer SAP-HANA-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Konfigurieren Sie den Tabellennamen, indem Sie angeben*tableName*.
3. Geben Sie unter Benutzerdefinierte Teradata-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines SAP-HANA-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie bei der Programmierung AWS Glue für Spark-Skripte.

Siehe the section called “SAP-HANA-Verbindungen”.

Verbindung zu SAP herstellen OData

SAP OData ist ein Standard-Webprotokoll, das für die Abfrage und Aktualisierung von in SAP vorhandenen Daten mithilfe von ABAP (Advanced Business Application Programming) verwendet wird. Dabei werden Webtechnologien wie HTTP angewendet und darauf aufgebaut, um den Zugriff auf Informationen aus einer Vielzahl von externen Anwendungen, Plattformen und Geräten zu ermöglichen. Mit dem Produkt können Sie auf alles zugreifen, was Sie für eine nahtlose Integration in Ihr SAP-System, Ihre Anwendung oder Ihre Daten benötigen.

Themen

- AWS Glue Unterstützung für SAP OData
- Verbindungen herstellen
- OData SAP-Job erstellen
- Verwenden des OData SAP-State-Management-Skripts
- Partitionierung für Nicht-ODP-Entitäten
- OData SAP-Verbindungsoptionen
- OData SAP-Entitäts- und Felddetails

AWS Glue Unterstützung für SAP OData

AWS Glue unterstützt SAP OData wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus SAP abzufragen OData.

Unterstützte OData SAP-API-Versionen

Die folgenden OData SAP-API-Versionen werden unterstützt:

- 2.0

Unterstützte Quellen

Die folgenden Quellen werden unterstützt:

- ODP-Quellen (Operational Data Provisioning):
  - BW-Extraktoren () DataSources
  - CDS-Ansichten
  - SALZ
- Nicht-ODP-Quellen, zum Beispiel:
  - CDS Dienste anzeigen
  - RFC-basierte Dienste
  - Benutzerdefinierte ABAP-Dienste

Unterstützte SAP-Komponenten

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie müssen den Katalogdienst für die Diensterkennung aktivieren.
  - Konfigurieren Sie ODP-Datenquellen (Operational Data Provisioning) für die Extraktion im SAP-Gateway Ihres SAP-Systems.
  - OData V2.0: Aktivieren Sie die OData V2.0-Katalogdienste in Ihrem SAP-Gateway per Transaktion. `/IWFND/MAINT_SERVICE`
  - Aktivieren Sie OData V2.0-Dienste in Ihrem SAP-Gateway per Transaktion. `/IWFND/MAINT_SERVICE`
  - Ihr OData SAP-Service muss clientseitige Paginierungs-/Abfrageoptionen wie und unterstützen. `$top` `$skip` Er muss auch die Systemabfrageoption unterstützen. `$count`
  - Sie müssen dem Benutzer in SAP die erforderliche Autorisierung zur Verfügung stellen, um die Dienste zu ermitteln und Daten mithilfe von OData SAP-Diensten zu extrahieren. Weitere Informationen finden Sie in der von SAP bereitgestellten Sicherheitsdokumentation.
- Wenn Sie OAuth 2.0 als Autorisierungsmechanismus verwenden möchten, müssen Sie OAuth 2.0 für den OData Service aktivieren und den OAuth Client gemäß der SAP-Dokumentation registrieren.
- Um einen OData Service auf der Grundlage von ODP-Datenquellen zu generieren, muss SAP Gateway Foundation lokal in Ihrem ERP/BW-Stack oder in einer Hub-Konfiguration installiert sein.

  - Für Ihre ERP/BW-Anwendungen muss der SAP NetWeaver AS ABAP-Stack die Version 7.50 SP02 oder höher haben.
  - Für das Hub-System (SAP Gateway) muss der SAP NetWeaver AS ABAP des Hub-Systems 7.50 SP01 oder höher sein, damit der Remote-Hub eingerichtet werden kann.
- Für Nicht-ODP-Quellen muss Ihre NetWeaver SAP-Stack-Version 7.40 SP02 oder höher sein.

Unterstützte Authentifizierungsmethoden

Die folgenden Authentifizierungsmethoden werden unterstützt:

- Standardauthentifizierung
- OAuth 2.0

Voraussetzungen

Bevor Sie einen AWS Glue Job für die Datenextraktion aus SAP OData über die OData SAP-Verbindung initiieren, müssen Sie die folgenden Voraussetzungen erfüllen:

- Der entsprechende OData SAP-Service muss im SAP-System aktiviert sein, um sicherzustellen, dass die Datenquelle zur Nutzung verfügbar ist. Wenn der OData Service nicht aktiviert ist, kann der Glue-Job nicht auf Daten aus SAP zugreifen oder diese extrahieren.
- Um sicherzustellen, dass der AWS Glue Job erfolgreich eine Verbindung mit dem OData SAP-Service herstellen kann, müssen geeignete Authentifizierungsmechanismen wie die (benutzerdefinierte) Standardauthentifizierung oder OAuth 2.0 in SAP konfiguriert werden.
- Konfigurieren Sie IAM-Richtlinien, um dem AWS Glue Job die entsprechenden Berechtigungen für den Zugriff auf SAP, Secrets Manager und andere am Prozess beteiligte AWS Ressourcen zu gewähren.
- Wenn das SAP-System in einem privaten Netzwerk gehostet wird, muss die VPC-Konnektivität so konfiguriert werden, dass der AWS Glue Job sicher mit SAP kommunizieren kann, ohne sensible Daten über das öffentliche Internet preiszugeben.

AWS Secrets Manager kann verwendet werden, um vertrauliche Informationen wie SAP-Anmeldeinformationen sicher zu speichern, die der AWS Glue Job zur Laufzeit dynamisch abrufen kann. Durch diesen Ansatz entfällt die Notwendigkeit, Anmeldeinformationen fest zu codieren, was die Sicherheit und Flexibilität erhöht.

Die folgenden Voraussetzungen geben step-by-step Hinweise zur Einrichtung der einzelnen Komponenten für eine reibungslose Integration zwischen AWS Glue und SAP OData.

Themen

- OData SAP-Aktivierung
- IAM-Richtlinien
- Konnektivität//VPC-Verbindung
- SAP-Authentifizierung
- AWS Secrets Manager zum Speichern Ihres Authentifizierungsgeheimnisses

OData SAP-Aktivierung

Führen Sie die folgenden Schritte für die OData SAP-Verbindung aus:

ODP-Quellen

Bevor Sie Daten von einem ODP-Anbieter übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

- Sie haben eine SAP NetWeaver AS ABAP-Instanz.
- Ihre NetWeaver SAP-Instanz enthält einen ODP-Provider, von dem Sie Daten übertragen möchten. Zu den ODP-Anbietern gehören:
  - SAP DataSources (Transaktionscode) RSO2
  - ABAP-CDS-Ansichten von SAP Core Data Services
  - SAP BW- oder SAP BW/4HANA-Systeme (, Objekt) InfoObject DataStore
  - Echtzeit-Replikation von Tabellen und DB-Views aus dem SAP-Quellsystem über den SAP Landscape Replication Server (SAP SLT)
  - SAP HANA-Informationsansichten in SAP-ABAP-basierten Quellen
- Ihre NetWeaver SAP-Instanz hat die Komponente SAP Gateway Foundation.
- Sie haben einen OData Dienst erstellt, der Daten von Ihrem ODP-Anbieter extrahiert. Um den OData Service zu erstellen, verwenden Sie den SAP Gateway Service Builder. Um auf Ihre ODP-Daten zuzugreifen, AppFlow ruft Amazon diesen Service mithilfe der OData API auf. Weitere Informationen finden Sie unter Generieren eines Dienstes zum Extrahieren von ODP-Daten über OData in der SAP BW/4HANA-Dokumentation.

- Um einen OData Service auf der Grundlage von ODP-Datenquellen zu generieren, muss SAP Gateway Foundation lokal in Ihrem ERP/BW-Stack oder in einer Hub-Konfiguration installiert sein.
  - Für Ihre ERP/BW-Anwendungen muss der SAP NetWeaver AS ABAP-Stack die Version 7.50 SP02 oder höher haben.
  - Für das Hub-System (SAP Gateway) muss der SAP NetWeaver AS ABAP des Hub-Systems 7.50 SP01 oder höher sein, damit der Remote-Hub eingerichtet werden kann.

Nicht-ODP-Quellen

- Ihre NetWeaver SAP-Stack-Version muss 7.40 SP02 oder höher sein.
- Sie müssen den Katalogdienst für die Diensterkennung aktivieren.
  - OData V2.0: Die Katalogdienste der OData Version 2.0 können in Ihrem SAP-Gateway per Transaktion aktiviert werden `/IWFND/MAINT_SERVICE`
- Ihr OData SAP-Service muss clientseitige Paginierungs-/Abfrageoptionen wie und unterstützen. `$top` `$skip` Er muss auch die Systemabfrageoption unterstützen. `$count`
- Für OAuth 2.0 müssen Sie OAuth 2.0 für den OData Service aktivieren und den OAuth Client gemäß der SAP-Dokumentation registrieren und die autorisierte Umleitungs-URL wie folgt festlegen:
  - `https://<region>.console.aws.amazon.com/gluestudio/oauth`, `<region>` ersetzt durch die Region, in der ausgeführt AWS Glue wird, Beispiel: us-east-1.
  - Sie müssen die sichere Einrichtung für die Verbindung über HTTPS aktivieren.
- Sie müssen dem Benutzer in SAP die erforderliche Autorisierung zur Verfügung stellen, um die Dienste zu ermitteln und Daten mithilfe von OData SAP-Services zu extrahieren. Weitere Informationen finden Sie in der von SAP bereitgestellten Sicherheitsdokumentation.

IAM-Richtlinien

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue"
            ],
            "Resource": "*"
         }
    ]
}
```

Die Rolle muss Zugriff auf alle vom Job verwendeten Ressourcen gewähren, z. B. Amazon S3. Wenn Sie die oben genannte Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien.

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

- SecretsManagerReadWrite— Ermöglicht Lese-/Schreibzugriff auf AWS Secrets Manager über die AWS Management Console. Hinweis: Dies schließt IAM-Aktionen aus. Kombinieren Sie es daher mit, `IAMFullAccess` wenn eine Rotationskonfiguration erforderlich ist.

IAM-Richtlinien/Berechtigungen, die für die Konfiguration von VPC erforderlich sind

Die folgenden IAM-Berechtigungen sind erforderlich, wenn Sie die VPC-Verbindung zum Erstellen einer AWS Glue Verbindung verwenden. Weitere Informationen finden Sie unter Erstellen einer IAM-Richtlinie für. AWS Glue

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Konnektivität//VPC-Verbindung

Schritte für die VPC-Verbindung:

1. Verwenden Sie eine bestehende VPC-Verbindung oder erstellen Sie eine neue Verbindung, indem Sie der Amazon VPC-Dokumentation folgen.
2. Stellen Sie sicher, dass Sie über ein NAT-Gateway verfügen, das den Datenverkehr ins Internet weiterleitet.
3. Wählen Sie VPC-Endpunkt als Amazon S3 S3-Gateway, um eine Verbindung herzustellen.
4. Aktivieren Sie die DNS-Auflösung und den DNS-Hostnamen, um die AWS bereitgestellten DNS-Dienste zu nutzen.

5. Gehen Sie zur erstellten VPC und fügen Sie die erforderlichen Endpunkte für verschiedene Dienste wie STS AWS Glue, Secret Managers hinzu.
   a. Klicken Sie auf Endpunkt erstellen.
   b. Wählen Sie als Servicekategorie die Option AWS Dienste aus.
   c. Wählen Sie als Dienstname den Dienst aus, mit dem Sie eine Verbindung herstellen.
   d. Wählen Sie VPC und Enable DNS Name.
   e. Für die VPC-Verbindung erforderliche VCP-Endpunkte:
      i. STS
      ii. AWS Glue
      iii. Secrets Manager

Sicherheitsgruppenkonfiguration

Die Sicherheitsgruppe muss Datenverkehr von der AWS Glue VPC zu ihrem Listening-Port zulassen AWS Glue , um eine Verbindung zu ihr herstellen zu können. Es empfiehlt sich, den Bereich der Quell-IP-Adressen so weit wie möglich einzuschränken.

AWS Glue erfordert eine spezielle Sicherheitsgruppe, die den gesamten eingehenden Datenverkehr von selbst zulässt. Sie können eine selbstreferenzierende Regel erstellen, die den gesamten von der Sicherheitsgruppe ausgehenden Datenverkehr zulässt. Sie können eine bestehende Sicherheitsgruppe ändern und die Sicherheitsgruppe als Quelle angeben.

Öffnen Sie die Kommunikation über die HTTPS-Ports des URL-Endpunkts (entweder NLB- oder SAP-Instanz).

Konnektivitätsoptionen

- HTTPS-Verbindung mit interner und externer NLB, SSL-Zertifikat von der Zertifizierungsstelle (CA), kein selbstsigniertes SSL-Zertifikat
- HTTPS-Verbindung mit SSL-Zertifikat der SAP-Instanz von der Zertifizierungsstelle (CA), kein selbstsigniertes SSL-Zertifikat

SAP-Authentifizierung

Der SAP-Konnektor unterstützt sowohl CUSTOM- (das ist SAP BASIC-Authentifizierung) als auch OAUTH-Authentifizierungsmethoden.

Benutzerdefinierte Authentifizierung

AWS Glue unterstützt Custom (Basic Authentication) als Methode zum Herstellen von Verbindungen zu Ihren SAP-Systemen und ermöglicht die Verwendung eines Benutzernamens und eines Kennworts für den sicheren Zugriff. Dieser Authentifizierungstyp eignet sich gut für Automatisierungsszenarien, da er die Verwendung von Benutzername und Passwort im Voraus mit den Berechtigungen eines bestimmten Benutzers in der OData SAP-Instanz ermöglicht. AWS Glue ist in der Lage, den Benutzernamen und das Passwort zur Authentifizierung von SAP zu verwenden. OData APIs In wird AWS Glue die Basisautorisierung als benutzerdefinierte Autorisierung implementiert.

Die öffentliche OData SAP-Dokumentation für Basic Auth Flow finden Sie unter HTTP Basic Authentication.

OAuth 2.0 Authentifizierung

AWS Glue unterstützt auch OAuth 2.0 als sicheren Authentifizierungsmechanismus für den Aufbau von Verbindungen zu Ihren SAP-Systemen. Dies ermöglicht eine nahtlose Integration und gewährleistet gleichzeitig die Einhaltung moderner Authentifizierungsstandards und erhöht die Sicherheit des Datenzugriffs.

AUTHORIZATION_CODE Art des Zuschusses

Die Art der Gewährung bestimmt, wie mit SAP AWS Glue kommuniziert wird, um Zugriff OData auf Ihre Daten anzufordern. SAP OData unterstützt nur den `AUTHORIZATION_CODE` Zuschusstyp. Dieser Zuschusstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue

Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in SAP zu erstellen OData und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu SAP weitergeleitet, OData um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken.
Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.

Eine öffentliche OData SAP-Dokumentation zur Erstellung einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter Authentifizierung mit OAuth 2.0.

AWS Secrets Manager zum Speichern Ihres Authentifizierungsgeheimnisses

Sie müssen die OData SAP-Verbindungsgeheimnisse im AWS Secrets Manager speichern, die erforderlichen Zugriffsberechtigungen für den Abruf konfigurieren, wie im IAM-Richtlinien Abschnitt angegeben, und sie beim Erstellen einer Verbindung verwenden.

Verwenden Sie die AWS Management Console for AWS Secrets Manager, um ein Geheimnis für Ihre SAP-Quelle zu erstellen. Weitere Informationen finden Sie unter Erstellen eines AWS Secrets Manager Manager-Geheimnisses. Die Details in AWS Secrets Manager sollten die Elemente im folgenden Code enthalten.

Benutzerdefiniertes Authentifizierungsgeheimnis

Sie müssen Ihren SAP-Systembenutzernamen anstelle von <your SAP username>und das zugehörige Passwort anstelle von <your SAP username password>True oder False eingeben. In diesem Zusammenhang `true` deaktiviert die Einstellung `basicAuthDisableSSO` auf Single Sign-On (SSO) für Anfragen zur Standardauthentifizierung, sodass für jede Anfrage explizite Benutzeranmeldedaten erforderlich sind. Umgekehrt `false` ermöglicht die Einstellung auf die Verwendung vorhandener SSO-Sitzungen, sofern verfügbar.

```
{
   "basicAuthUsername": "<your SAP username>",
   "basicAuthPassword": "<your SAP username password>",
   "basicAuthDisableSSO": "<True/False>",
   "customAuthenticationType": "CustomBasicAuth"
}
```

OAuth 2.0 Geheim

Falls Sie OAuth 2.0 als Authentifizierungsmechanismus verwenden, sollte das Geheimnis im AWS Secrets Manager die User Managed Client Application ClientId im folgenden Format haben. Sie müssen stattdessen Ihr SAP-Clientgeheimnis eingeben<your client secret>.

```
{"USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET": "<your client secret>"
}
```

Verbindungen herstellen

Um eine OData SAP-Verbindung zu konfigurieren:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die AWS Glue Konsole. Stellen Sie im AWS Glue Studio eine Verbindung her, indem Sie die folgenden Schritte ausführen:
   a. Klicken Sie im linken Bereich auf Datenverbindungen.
   b. Klicken Sie auf Verbindung erstellen.
   c. Wählen Sie SAP OData unter Datenquelle auswählen
   d. Geben Sie die Anwendungshost-URL der OData SAP-Instanz an, zu der Sie eine Verbindung herstellen möchten. Diese Host-URL der Anwendung muss über das öffentliche Internet zugänglich sein, wenn keine VPC-Verbindung besteht.
   e. Geben Sie den Anwendungsdienstpfad der OData SAP-Instanz an, zu der Sie eine Verbindung herstellen möchten. Dies entspricht dem Katalog-Servicepfad. Zum Beispiel:`/sap/opu/odata/iwfnd/catalogservice;v=2`. AWS Glue akzeptiert keinen bestimmten Objektpfad.
   f. Geben Sie die Kundennummer der OData SAP-Instanz an, zu der Sie eine Verbindung herstellen möchten. Zulässige Werte sind [001-999]. Beispiel: 010
   g. Geben Sie die Port-Nummer der OData SAP-Instanz an, zu der Sie eine Verbindung herstellen möchten. Beispiel: 443
   h. Geben Sie die Anmeldesprache der OData SAP-Instanz an, zu der Sie eine Verbindung herstellen möchten. Beispiel: DE
   i. Wählen Sie die AWS IAM-Rolle aus, die die im IAM-Richtlinien Abschnitt beschriebenen Berechtigungen übernehmen AWS Glue kann und über diese verfügt.
   j. Wählen Sie den Authentifizierungstyp, den Sie für diese Verbindung verwenden möchten, AWS Glue aus der Dropdownliste aus: oder BENUTZERDEFINIERT OAUTH2
      i. BENUTZERDEFINIERT — Wählen Sie das Geheimnis aus, das Sie wie im AWS Secrets Manager zum Speichern Ihres Authentifizierungsgeheimnisses Abschnitt angegeben erstellt haben.
      ii. OAUTH 2.0 — geben Sie die folgenden Eingaben nur bei 2.0 ein: OAuth
         A. Geben Sie unter Benutzerverwaltete Client-Anwendung ClientId Ihre Client-ID ein.
         B. `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET`(Ihr Client-Geheimnis) im AWS Secrets Manager, den Sie in dem AWS Secrets Manager zum Speichern Ihres Authentifizierungsgeheimnisses Abschnitt erstellt haben.
         C. Geben Sie unter Autorisierungscode-URL Ihre Autorisierungscode-URL ein.

D. Geben Sie unter Autorisierungstoken-URL Ihre Autorisierungstoken-URL ein.

E. Geben Sie unter OAuth Bereiche Ihre OAuth Bereiche durch Leerzeichen getrennt ein. Beispiel: `/IWFND/SG_MED_CATALOG_0002 ZAPI_SALES_ORDER_SRV_0001`

k. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten. Weitere Details finden Sie unter Konnektivität//VPC-Verbindung.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte. Weitere Informationen hierzu finden Sie unter IAM-Richtlinien.
3. Wählen Sie Verbindung testen und testen Sie Ihre Verbindung. Wenn der Verbindungstest erfolgreich ist, klicken Sie auf Weiter, geben Sie Ihren Verbindungsnamen ein und speichern Sie Ihre Verbindung. Die Testverbindungsfunktion ist nicht verfügbar, wenn Sie Netzwerkoptionen (VPC) ausgewählt haben.

OData SAP-Job erstellen

Weitere Informationen finden Sie unter Visual ETL-Jobs mit AWS Glue Studio erstellen

Quellen für Operational Data Provisioning (ODP)

Operational Data Provisioning (ODP) bietet eine technische Infrastruktur, die Sie zur Unterstützung der Datenextraktion und Replikation für verschiedene Zielanwendungen verwenden können, und unterstützt Deltamechanismen in diesen Szenarien. Bei einem Delta-Verfahren werden die Daten aus einer Quelle (ODP-Provider) mithilfe eines Aktualisierungsprozesses automatisch in eine Delta-Queue (Operational Delta Queue — ODQ) geschrieben oder über eine Extraktorschnittstelle an die Delta-Warteschlange übergeben. Ein ODP-Anbieter kann ein DataSource (Extraktor), ABAP Core Data Services Views (ABAP CDS Views), SAP BW oder SAP BW/4HANA, SAP Landscape Transformation Replication Server (SLT) und SAP HANA Information Views (Berechnungsansichten) sein. Die Zielanwendungen (als ODQ-Abonnenten oder allgemeiner als „ODP-Verbraucher“ bezeichnet) rufen die Daten aus der Delta-Warteschlange ab und setzen die Verarbeitung der Daten fort.

Volllast

Im Kontext von SAP OData - und ODP-Entitäten bezieht sich ein Full Load auf den Vorgang, bei dem alle verfügbaren Daten aus einer ODP-Entität in einem einzigen Vorgang extrahiert werden. Dieser Vorgang ruft den vollständigen Datensatz aus dem Quellsystem ab und stellt so sicher, dass das Zielsystem über eine umfassende up-to-date Kopie der Daten der Entität verfügt. Vollständige

Ladevorgänge werden in der Regel für Quellen verwendet, die inkrementelle Ladevorgänge nicht unterstützen, oder wenn eine Aktualisierung des Zielsystems erforderlich ist.

Beispiel

Sie können das ENABLE_CDC Flag explizit auf False setzen, wenn Sie das DynamicFrame erstellen. Hinweis: ENABLE_CDC ist standardmäßig falsch. Wenn Sie die Delta-Warteschlange nicht initialisieren möchten, müssen Sie dieses Flag nicht senden oder auf true setzen. Wenn Sie dieses Flag nicht auf true setzen, führt dies zu einer vollständigen Ladeextraktion.

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "false"
    }, transformation_ctx=key)
```

Inkrementelles Laden

Ein inkrementelles Laden im Kontext von ODP-Entitäten (Operational Data Provisioning) beinhaltet das Extrahieren nur der neuen oder geänderten Daten (Deltas) aus dem Quellsystem seit der letzten Datenextraktion, wodurch eine Vorverarbeitung der bereits verarbeiteten Datensätze vermieden wird. Dieser Ansatz verbessert die Effizienz erheblich, reduziert das Datenübertragungsvolumen, verbessert die Leistung, gewährleistet eine effiziente Synchronisation zwischen Systemen und minimiert die Verarbeitungszeit, insbesondere bei großen Datensätzen, die sich häufig ändern.

Inkrementelle Transfers auf Delta-Token-Basis

Gehen Sie wie folgt vor, um die inkrementelle Übertragung mithilfe von Change Data Capture (CDC) für ODP-fähige Entitäten zu aktivieren, die dies unterstützen:

1. Erstellen Sie den Auftrag „Inkrementelle Übertragung“ im Skriptmodus.
2. Beim Erstellen von DataFrame oder DynamicFrame Glue müssen Sie die Option übergeben`"ENABLE_CDC": "True"`. Diese Option stellt sicher, dass Sie von SAP ein Delta-Token erhalten, das für den späteren Abruf geänderter Daten verwendet werden kann.

Das Delta-Token wird in der letzten Zeile des Datenrahmens in der Spalte DELTA_TOKEN vorhanden sein. Dieses Token kann bei nachfolgenden Aufrufen als Konnektoroption verwendet werden, um den nächsten Datensatz inkrementell abzurufen.

Beispiel

- Wir setzen die ENABLE_CDC Flagge bei `true` der Erstellung von auf DynamicFrame. Hinweis: ENABLE_CDC Ist `false` standardmäßig. Wenn Sie die Delta-Warteschlange nicht initialisieren möchten, müssen Sie dieses Flag nicht senden oder es auf true setzen. Wenn Sie dieses Flag nicht auf true setzen, führt dies zu einer vollständigen Ladeextraktion.

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true"
    }, transformation_ctx=key)

# Extract the delta token from the last row of the DELTA_TOKEN column
delta_token_1 = your_logic_to_extract_delta_token(sapodata_df) # e.g.,
 D20241029164449_000370000
```

- Das extrahierte Delta-Token kann als Option zum Abrufen neuer Ereignisse übergeben werden.

```
sapodata_df_2 = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        // passing the delta token retrieved in the last run
        "DELTA_TOKEN": delta_token_1
    } , transformation_ctx=key)

# Extract the new delta token for the next run
delta_token_2 = your_logic_to_extract_delta_token(sapodata_df_2)
```

Beachten Sie, dass der letzte Datensatz, in dem der vorhanden DELTA_TOKEN ist, kein Transaktionsdatensatz aus der Quelle ist und nur dazu dient, den Delta-Token-Wert zu übergeben.

Abgesehen von den DELTA_TOKEN werden die folgenden Felder in jeder Zeile des Datenrahmens zurückgegeben.

- GLUE_FETCH_SQ: Dies ist ein Sequenzfeld, das aus dem EPOC-Zeitstempel in der Reihenfolge generiert wird, in der der Datensatz empfangen wurde, und das für jeden Datensatz eindeutig ist.

Dies kann verwendet werden, wenn Sie die Reihenfolge der Änderungen im Quellsystem kennen oder festlegen müssen. Dieses Feld ist nur für ODP-fähige Entitäten vorhanden.

- DML_STATUS: Dies wird `UPDATED` für alle neu eingefügten und aktualisierten Datensätze aus der Quelle und `DELETED` für Datensätze angezeigt, die aus der Quelle gelöscht wurden.

Weitere Informationen zur Verwaltung des Status und zur Wiederverwendung des Delta-Tokens zum Abrufen geänderter Datensätze anhand eines Beispiels finden Sie im Abschnitt. Verwenden des OData SAP-State-Management-Skripts

Invalidierung des Delta-Tokens

Ein Delta-Token ist der Servicesammlung und einem Benutzer zugeordnet. Wenn ein neuer initialer Pull mit `“ENABLE_CDC” : “true”` für dieselbe Servicesammlung und denselben Benutzer initiiert wird, werden alle vorherigen Delta-Token, die als Ergebnis einer vorherigen Initialisierung ausgegeben wurden, vom OData SAP-Service ungültig gemacht. Das Aufrufen des Connectors mit einem abgelaufenen Delta-Token führt zu einer Ausnahme:

```
Could not open data access via extraction API RODPS_REPL_ODP_OPEN
```

OData Dienste (Nicht-ODP-Quellen)

Volle Ladung

Bei Systemen ohne ODP (Operational Data Provisioning) wird beim Vollladen der gesamte Datensatz aus dem Quellsystem extrahiert und in das Zielsystem geladen. Da Nicht-ODP-Systeme von Haus aus keine fortschrittlichen Datenextraktionsmechanismen wie Deltas unterstützen, ist der Vorgang unkompliziert, kann aber je nach Datengröße ressourcenintensiv sein.

Inkrementelles Laden

Bei Systemen oder Entitäten, die ODP (Operational Data Provisioning) nicht unterstützen, kann die inkrementelle Datenübertragung manuell verwaltet werden, indem ein auf Zeitstempeln basierender Mechanismus zur Nachverfolgung und Extraktion von Änderungen implementiert wird.

Auf Zeitstempeln basierende inkrementelle Übertragungen

Für nicht ODP-fähige Entitäten (oder für ODP-fähige Entitäten, die das ENABLE_CDC-Flag nicht verwenden) können wir eine `filteringExpression` Option im Konnektor verwenden, um das Intervall anzugeben, für das wir Daten abrufen möchten. `datetime` Diese Methode basiert auf einem Zeitstempelfeld in Ihren Daten, das angibt, wann jeder Datensatz zuletzt erstellt/geändert wurde.

Beispiel

Datensätze werden abgerufen, die sich nach 2024-01-01T 00:00:00.000 geändert haben

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "filteringExpression": "LastChangeDateTime >= 2024-01-01T00:00:00.000"
    }, transformation_ctx=key)
```

Hinweis: In diesem Beispiel `LastChangeDateTime` ist dies das Feld, das angibt, wann jeder Datensatz zuletzt geändert wurde. Der tatsächliche Feldname kann je nach Ihrer spezifischen OData SAP-Entität variieren.

Um in nachfolgenden Durchläufen eine neue Teilmenge von Daten zu erhalten, würden Sie die `filteringExpression` mit einem neuen Zeitstempel aktualisieren. In der Regel wäre dies der maximale Zeitstempelwert der zuvor abgerufenen Daten.

Beispiel

```
max_timestamp = get_max_timestamp(sapodata_df)  # Function to get the max timestamp
 from the previous run
next_filtering_expression = f"LastChangeDateTime > {max_timestamp}"

# Use this next_filtering_expression in your next run
```

Im nächsten Abschnitt stellen wir einen automatisierten Ansatz zur Verwaltung dieser auf Zeitstempeln basierenden inkrementellen Übertragungen vor, sodass der Filterausdruck zwischen den Durchläufen nicht mehr manuell aktualisiert werden muss.

Verwenden des OData SAP-State-Management-Skripts

Gehen Sie folgendermaßen vor, um das OData SAP-State-Management-Skript in Ihrem AWS Glue Job zu verwenden:

- Laden Sie das State-Management-Skript: `s3://aws-blogs-artifacts-public/artifacts/BDB-4789/sap_odata_state_management.zip` aus dem öffentlichen Amazon S3 S3-Bucket herunter.
- Laden Sie das Skript in einen Amazon S3 S3-Bucket hoch, für den Ihr AWS Glue Job Zugriffsberechtigungen besitzt.
- Referenzieren Sie das Skript in Ihrem AWS Glue Job: Wenn Sie Ihren AWS Glue Job erstellen oder aktualisieren, übergeben Sie die `'--extra-py-files'` Option, die auf den Skriptpfad in Ihrem Amazon S3 S3-Bucket verweist. Beispiel: `--extra-py-files s3://your-bucket/path/to/sap_odata_state_management.py`
- Importieren und verwenden Sie die State Management Library in Ihren AWS Glue Job-Skripten.

Beispiel für eine inkrementelle Übertragung auf Delta-Token-Basis

Hier ist ein Beispiel für die Verwendung des State-Management-Skripts für inkrementelle Übertragungen auf Delta-Token-Basis:

```
from sap_odata_state_management import StateManagerFactory, StateManagerType, StateType

# Initialize the state manager
state_manager = StateManagerFactory.create_manager(
    manager_type=StateManagerType.JOB_TAG,
    state_type=StateType.DELTA_TOKEN,
    options={
        "job_name": args['JOB_NAME'],
        "logger": logger
    }
)

# Get connector options (including delta token if available)
key = "SAPODataNode"
connector_options = state_manager.get_connector_options(key)

# Use the connector options in your Glue job
df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true",
        **connector_options
```

```
    }
)

# Process your data here...

# Update the state after processing
state_manager.update_state(key, sapodata_df.toDF())
```

Beispiel für eine auf Zeitstempeln basierende inkrementelle Übertragung

Hier ist ein Beispiel für die Verwendung des State-Management-Skripts für inkrementelle Übertragungen auf Delta-Token-Basis:

```
from sap_odata_state_management import StateManagerFactory, StateManagerType, StateType

# Initialize the state manager
state_manager = StateManagerFactory.create_manager(
    manager_type=StateManagerType.JOB_TAG,
    state_type=StateType.DELTA_TOKEN,
    options={
        "job_name": args['JOB_NAME'],
        "logger": logger
    }
)

# Get connector options (including delta token if available)
key = "SAPODataNode"
connector_options = state_manager.get_connector_options(key)

# Use the connector options in your Glue job
df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true",
        **connector_options
    }
)

# Process your data here...

# Update the state after processing
```

```
state_manager.update_state(key, sapodata_df.toDF())
```

In beiden Beispielen bewältigt das State-Management-Skript die Komplexität der Speicherung des Status (entweder Delta-Token oder Zeitstempel) zwischen Auftragsausführungen. Es ruft beim Abrufen der Connector-Optionen automatisch den letzten bekannten Status ab und aktualisiert den Status nach der Verarbeitung, sodass sichergestellt wird, dass bei jeder Ausführung des Jobs nur neue oder geänderte Daten verarbeitet werden.

Partitionierung für Nicht-ODP-Entitäten

In Apache Spark bezieht sich Partitionierung auf die Art und Weise, wie Daten aufgeteilt und auf die Worker-Knoten in einem Cluster zur Parallelverarbeitung verteilt werden. Jede Partition ist ein logischer Datenblock, der unabhängig von einer Aufgabe verarbeitet werden kann. Partitionierung ist ein grundlegendes Konzept in Spark, das sich direkt auf Leistung, Skalierbarkeit und Ressourcennutzung auswirkt. AWS Glue Jobs verwenden den Partitionierungsmechanismus von Spark, um den Datensatz in kleinere Blöcke (Partitionen) aufzuteilen, die parallel über die Worker-Knoten des Clusters verarbeitet werden können. Beachten Sie, dass die Partitionierung nicht für ODP-Entitäten gilt.

Weitere Informationen finden Sie unter AWS Glue Spark und PySpark Jobs.

Voraussetzungen

Ein OData SAP-Objekt, aus dem Sie lesen möchten. Sie benötigen zum Beispiel das EntitySet Objekt/den Namen. `/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder`

Beispiel

```
sapodata_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder"
    }, transformation_ctx=key)
```

Abfragen partitionieren

Feldbasierte Partitionierung

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`, und angeben`UPPER_BOUND`, `NUM_PARTITIONS` ob Sie Parallelität in Spark nutzen möchten. Mit diesen

Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können. Integer, Date und DateTime Felder unterstützen die feldbasierte Partitionierung im SAP-Konnektor. OData

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für jedes Feld, dessen Datentyp ist DateTime, wird das in Spark-SQL-Abfragen verwendete Spark-Zeitstempelformat akzeptiert.

  Beispiele für gültige Werte: "2000-01-01T00:00:00.000Z"
- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: Anzahl der Partitionen.
- PARTITION_BY: der Typ der durchzuführenden Partitionierung, FIELD der im Falle einer feldbasierten Partitionierung übergeben werden muss.

Beispiel

```
sapodata= glueContext.create_dynamic_frame.from_options(
    connection_type="sapodata",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/SEPM_HCM_SCENARIO_SRV/EmployeeSet",
        "PARTITION_FIELD": "validStartDate"
        "LOWER_BOUND": "2000-01-01T00:00:00.000Z"
        "UPPER_BOUND": "2020-01-01T00:00:00.000Z"
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "FIELD"
    }, transformation_ctx=key)
```

Datensatzbasierte Partitionierung

Die ursprüngliche Abfrage würde in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Die datensatzbasierte Partitionierung wird nur für Nicht-ODP-Entitäten unterstützt, da die Paginierung in ODP-Entitäten bis zum nächsten Token/Skip-Token unterstützt wird.

- PARTITION_BY: der Typ der durchzuführenden Partitionierung. COUNTist im Falle einer datensatzbasierten Partitionierung zu übergeben.

Beispiel

```
sapodata= glueContext.create_dynamic_frame.from_options(
    connection_type="sapodata",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/SEPM_HCM_SCENARIO_SRV/EmployeeSet",
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "COUNT"
    }, transformation_ctx=key)
```

Einschränkungen//Callouts

- ODP-Entitäten sind nicht mit der datensatzbasierten Partitionierung kompatibel, da die Paginierung mit Skip-Token/Delta-Token abgewickelt wird. Folglich wird für die datensatzbasierte Partitionierung der Standardwert für MaxConcurrency unabhängig von der Benutzereingabe auf „Null“ gesetzt.
- Wenn sowohl das Limit als auch die Partition angewendet werden, hat das Limit Vorrang vor der Partitionierung.

OData SAP-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für SAP aufgeführt OData:

- ENTITY_NAME(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in SAP OData.

  Zum Beispiel:/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_ SalesOrder
- API_VERSION(Zeichenfolge) — (Optional) Wird zum Lesen verwendet. Die OData SAP-Rest-API-Version, die Sie verwenden möchten. Beispiel: 2.0.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.

  Beispiel: SalesOrder

- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.

  Zum Beispiel: `SalesOrder = "10"`
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

  Zum Beispiel: `SELECT * FROM /sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder`
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.

  Zum Beispiel: `ValidStartDate`
- LOWER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Zum Beispiel: `"2000-01-01T00:00:00.000Z"`
- UPPER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.

  Zum Beispiel: `"2024-01-01T00:00:00.000Z"`
- NUM_PARTITIONS(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- INSTANCE_URL(String) — Die Host-URL der SAP-Instanzanwendung.

  Zum Beispiel: `https://example-externaldata.sierra.aws.dev`
- SERVICE_PATH(String) — Der Dienstpfad für die SAP-Instanzanwendung.

  Zum Beispiel: `/sap/opu/odata/iwfnd/catalogservice;v=2`
- CLIENT_NUMBER(String) — Die Kundennummer der SAP-Instanzanwendung.

  Zum Beispiel: 100
- PORT_NUMBER(String) — Standard: Die Portnummer der SAP-Instanzanwendung.

  Zum Beispiel: 443
- LOGON_LANGUAGE(String) — Die Anmeldesprache der SAP-Instanzanwendung.

  Zum Beispiel: `EN`

- ENABLE_CDC(String) — Definiert, ob ein Job mit aktiviertem CDC ausgeführt werden soll, d. h. mit Änderungen nachverfolgen.

  Zum Beispiel: True/False

- DELTA_TOKEN(String) — Führt einen inkrementellen Datenabruf auf der Grundlage des angegebenen gültigen Delta-Tokens aus.

  Zum Beispiel: D20241107043437_000463000

- PAGE_SIZE(Integer) — Definiert die Seitengröße für die Abfrage der Datensätze. Die Standard-Seitengröße ist 50.000. Wenn eine Seitengröße angegeben ist, gibt SAP nur die definierte Anzahl von Datensätzen pro API-Aufruf zurück, nicht den gesamten Datensatz. Der Connector stellt weiterhin die Gesamtzahl der Datensätze bereit und verarbeitet die Seitennummerierung unter Verwendung der von Ihnen angegebenen Seitengröße. Wenn Sie eine größere Seitengröße benötigen, können Sie einen beliebigen Wert bis zu 500.000 wählen, was der maximal zulässige Wert ist. Jede angegebene Seitengröße, die 500.000 überschreitet, wird ignoriert. Stattdessen verwendet das System die maximal zulässige Seitengröße. Sie können die Seitengröße in der AWS Glue Studio Benutzeroberfläche angeben, indem Sie eine Verbindungsoption PAGE_SIZE mit dem gewünschten Wert hinzufügen.

  Zum Beispiel: 20000

OData SAP-Entitäts- und Felddetails

Entität	Datentyp	Unterstützte Operatoren
Tabellen (dynamische Entitäten)	String	=,! =, >, >=, <, <=, ZWISCHEN, WIE
	Ganzzahl	=,! =, >, >=, <, <=, ZWISCHEN, WIE
	Long	=,! =, >, >=, <, <=, ZWISCHEN, WIE
	Double	=,! =, >, >=, <, <=, ZWISCHEN, WIE

Entität	Datentyp	Unterstützte Operatoren
	Datum	=,! =, >, >=, <, <=, ZWISCHEN, WIE
	DateTime	=,! =, >, >=, <, <=, ZWISCHEN, WIE
	Boolesch	=, !=
	Struct	=,! =, >, >=, <, <=, ZWISCHEN, WIE

Verbindung herstellen zu SendGrid

SendGrid ist eine Kundenkommunikationsplattform für Transaktions- und Marketing-E-Mails.

- SendGrid Connector hilft bei der Erstellung und Verwaltung von Kontaktlisten sowie bei der Erstellung von E-Mail-Marketingkampagnen.
- SendGrid ermöglicht es Online-Unternehmen, gemeinnützigen Organisationen und anderen Online-Organisationen, Marketing-E-Mails zu erstellen und an ein großes Publikum zu senden und die Interaktion mit diesen E-Mails zu überwachen.

Themen

- AWS Glue Unterstützung für SendGrid
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration SendGrid
- SendGrid Verbindungen konfigurieren
- Aus SendGrid Entitäten lesen
- SendGrid Verbindungsoptionen
- SendGrid Einschränkungen

AWS Glue Unterstützung für SendGrid

AWS Glue unterstützt SendGrid wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen SendGrid.

Wird es als Ziel unterstützt?

Nein.

Unterstützte SendGrid API-Versionen

Die folgenden SendGrid API-Versionen werden unterstützt:

- v3

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration SendGrid

Bevor Sie Daten von AWS Glue übertragen können SendGrid, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein SendGrid Konto mit einem API-Schlüssel.
- Ihr SendGrid Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem SendGrid Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun SendGrid.

SendGrid Verbindungen konfigurieren

SendGrid unterstützt benutzerdefinierte Authentifizierung.

Öffentliche SendGrid Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Authentifizierung finden Sie unter Authentifizierung.

So konfigurieren Sie eine SendGrid Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem *`api_key`* Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps SendGrid.
   b. Geben Sie die SendGrid Instanz `INSTANCE_URL` an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus SendGrid Entitäten lesen

Voraussetzung

Ein SendGrid Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie`lists`, `singlesends` oder`segments`.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Listen	Nein	Ja	Nein	Ja	Nein
Einzelne Sendungen	Ja	Ja	Nein	Ja	Nein
Statistiken und Automatisierungen von Marketingkampagnen	Ja	Ja	Nein	Ja	Nein
Statistiken zu Marketingkampagnen — einzelne Sendungen	Ja	Ja	Nein	Ja	Nein
Segmente	Ja	Nein	Nein	Ja	Nein
Kontakte	Ja	Nein	Nein	Ja	Nein
Kategorie	Nein	Nein	Nein	Ja	Nein
Statistiken	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Gruppen abbestellen	Ja	Nein	Nein	Ja	Nein

Beispiel:

```
sendgrid_read = glueContext.create_dynamic_frame.from_options(
    connection_type="sendgrid",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "lists",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
    }
```

SendGrid Entitäts- und Felddetails:

Entitäten mit statischen Metadaten:

Entität	Feld	Datentyp	Unterstützte Operatoren	
Listen	id	String	N/A	
	Name	String	N/A	
	contact_count	Ganzzahl	N/A	
	_metadaten	Struct	N/A	
Einzelne Sendungen	id	String	N/A	
	Name	String	EQUAL_TO	
	abscheulich	Struct	N/A	
	Status	String	EQUAL_TO	

Entität	Feld	Datentyp	Unterstützte Operatoren	
	categories	Auflisten	EQUAL_TO	
	send_at	String	N/A	
	ist_abtest	Boolesch	N/A	
	updated_at	String	N/A	
	created_at	String	N/A	
	Kanäle	Auflisten	N/A	
Statistiken und Automatisierungen von Marketingkampagnen	id	String	N/A	
	Aggregation	String	N/A	
	step_id	String	N/A	
	stats	Struct	N/A	
	Automatisierungs-IDs	Auflisten	EQUAL_TO	
Statistiken zur Marketingkampagne — Singlesends	id	String	N/A	
	ab_Variante	String	N/A	
	ab_phase	String	N/A	
	Aggregation	String	N/A	
	stats	Struct	N/A	
	singlesend_ids	Auflisten	EQUAL_TO	
Segmente	id	String	N/A	
	Name	String	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	
	Abfrageversion	String	N/A	
	Anzahl der Kontakte	Ganzzahl	N/A	
	Beispiel_aktualisiert_at	String	N/A	
	nächste_Beispielaktualisierung	String	N/A	
	created_at	String	N/A	
	updated_at	String	N/A	
	ID der übergeordneten Liste	String	N/A	
	Status	Struct	N/A	
	übergeordnete Listen-ID	String	EQUAL_TO	
	keine übergeordnete Listen-ID	Boolesch	EQUAL_TO	
Kontakte	id	String	N/A	
	first_name	String	N/A	
	last_name	String	N/A	
	eindeutiger_Name	String	N/A	
	email	String	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	
	alternate_E-Mails	Auflisten	N/A	
	address_line_1	String	N/A	
	address_line_2	String	N/A	
	city	String	N/A	
	Bundesstaat_Provinz/Region	String	N/A	
	country	String	N/A	
	postal_code	String	N/A	
	phone_number	String	N/A	
	WhatsApp	String	N/A	
	Linie	String	N/A	
	Facebook	String	N/A	
	IDs auflisten	Auflisten	N/A	
	custom_fields	Struct	N/A	
	created_at	String	N/A	
	updated_at	String	N/A	
	_metadaten	Struct	N/A	
	event_timestamp	DateTime	BETWEEN	
Kategorie	categories	Auflisten	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	
Statistiken	date	String	N/A	
	stats	Auflisten	N/A	
	Startdatum	DateTime	EQUAL_TO, ZWISCHEN	
	aggregated_by	String	EQUAL_TO	
Gruppen abbestellen	id	Ganzzahl	EQUAL_TO	
	Name	String	N/A	
	description	String	N/A	
	last_email_sent_at	Ganzzahl	N/A	
	ist_default	Boolesch	N/A	
	meldet sich ab	Ganzzahl	N/A	

> **Note**
>
> Die Datentypen Struct und List werden in den Datentyp String konvertiert, und der DateTime Datentyp wird in der Antwort der Konnektoren in Timestamp konvertiert.

Abfragen partitionieren

SendGrid unterstützt keine filterbasierte Partitionierung oder datensatzbasierte Partitionierung.

SendGrid Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für SendGrid:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in SendGrid.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. SendGrid Rest-API-Version, die Sie verwenden möchten.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige SendGrid Instanz-URL.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

SendGrid Einschränkungen

Im Folgenden finden Sie Einschränkungen oder Hinweise für SendGrid:

- Der inkrementelle Abruf wird nur von der Statistik-Entität auf dem `start_date` Feld und von der Kontakt-Entität auf dem `event_timestamp` Feld unterstützt.
- Die Paginierung wird nur von den Entitäten Marketingkampagnenstatistiken (Automatisierungen), Marketingkampagnenstatistiken (Einzelsendungen), Einzelsendungen und Listen unterstützt.
- Für die Entität Stats `start_date` ist dies ein obligatorischer Filterparameter.
- Ein API-Schlüssel mit eingeschränktem Zugriff kann keinen Lesezugriff für die E-Mail-API- und Statistik-Entitäten unterstützen. Verwenden Sie einen API-Schlüssel mit Vollzugriff. Weitere Informationen finden Sie unter API-Übersicht.

Verbindung herstellen zu ServiceNow

ServiceNow ist eine Cloud-basierte SaaS-Plattform zur Automatisierung von IT-Management-Workflows. Die ServiceNow Plattform lässt sich problemlos in andere Tools integrieren, sodass Benutzer Projekte, Teams und Kundeninteraktionen mithilfe einer Vielzahl von Apps und Plugins verwalten können. Als ServiceNow Benutzer können Sie eine Verbindung AWS Glue zu Ihrem ServiceNow Konto herstellen. Anschließend können Sie es ServiceNow als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten ServiceNow zwischen AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für ServiceNow
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration ServiceNow
- ServiceNow Verbindungen konfigurieren
- Aus ServiceNow Entitäten lesen
- ServiceNow Verbindungsoptionen
- Einschränkungen und Hinweise für ServiceNow Connector

AWS Glue Unterstützung für ServiceNow

AWS Glue unterstützt ServiceNow wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen ServiceNow.

Wird es als Ziel unterstützt?

Nein.

Unterstützte ServiceNow API-Versionen

Die folgenden ServiceNow API-Versionen werden unterstützt:

- v2

Informationen zur versionsspezifischen Unterstützung von Entitäten finden Sie unter Unterstützte Entitäten für Source.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
```

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration ServiceNow

Bevor Sie Daten von AWS Glue übertragen können ServiceNow, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein ServiceNow Konto mit E-Mail und Passwort. Weitere Informationen finden Sie unter Ein ServiceNow Konto erstellen.
- Ihr ServiceNow Konto ist für den API-Zugriff aktiviert. Jegliche Nutzung der ServiceNow API ist ohne zusätzliche Kosten möglich.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem ServiceNow Konto herstellen.

Ein ServiceNow Konto erstellen

Um ein ServiceNow Konto zu erstellen:

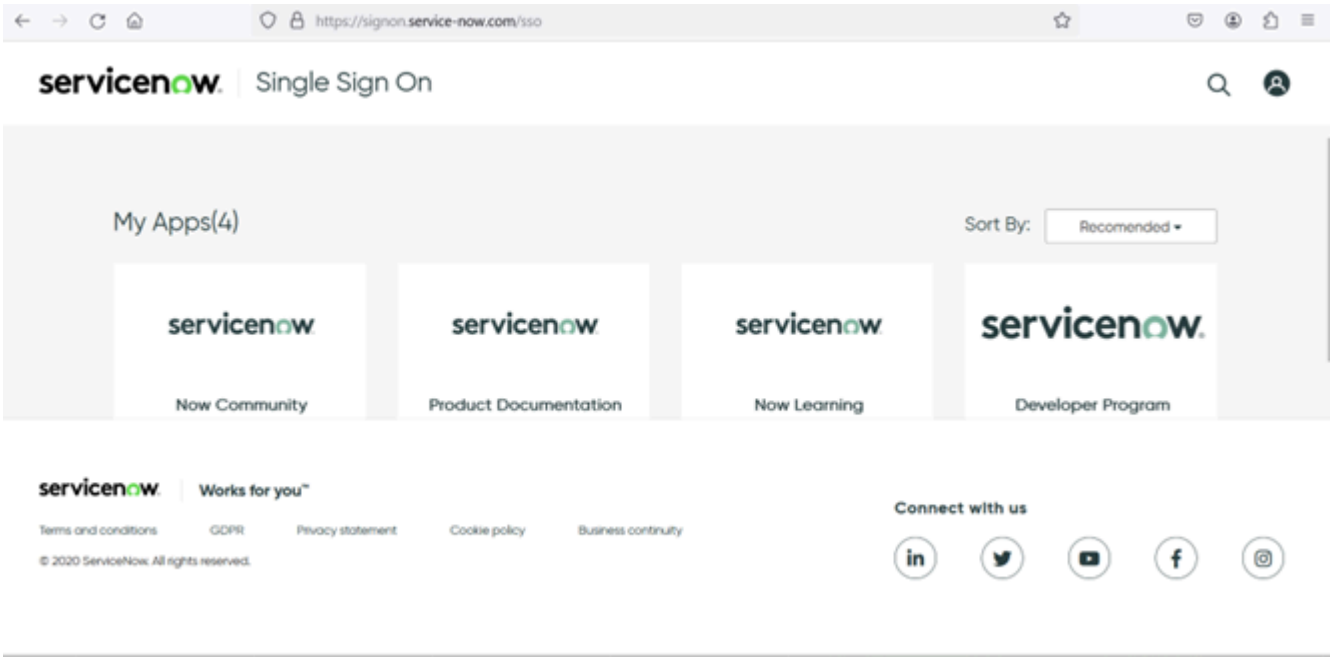
1. Gehen Sie zur Anmeldeseite auf servicenow.com, geben Sie Ihre Daten ein und klicken Sie auf Weiter.
2. Wenn Sie in Ihrer eingeschriebenen E-Mail einen Bestätigungscode erhalten, geben Sie diesen Code ein und wählen Sie Verifizieren.
3. Richten Sie die Multi-Faktor-Authentifizierung ein oder überspringen Sie dies.

Ihr Konto wurde erstellt und Ihr Profil ServiceNow wird angezeigt.

Eine ServiceNow Entwicklerinstanz erstellen

Fordern Sie eine Entwicklerinstanz an, nachdem Sie sich angemeldet haben ServiceNow.

1. Geben Sie auf der ServiceNow Anmeldeseite Ihre Kontoanmeldeinformationen ein.
2. Wählen Sie das ServiceNow Entwicklerprogramm.



3. Wählen Sie oben rechts die Option Request Instance aus.
4. Geben Sie Ihre beruflichen Verantwortlichkeiten ein. Geben Sie an, dass Sie mit den Nutzungsbedingungen einverstanden sind, und wählen Sie „Einrichtung beenden“.
5. Notieren Sie sich nach der Erstellung der Instanz Ihre Instanz-URL und Ihre Anmeldeinformationen.

Anmeldeinformationen werden abgerufen BasicAuth

So rufen Sie die Basic Auth-Anmeldeinformationen für ein kostenloses Konto ab:

1. Geben Sie auf der ServiceNow Anmeldeseite Ihre Kontoanmeldeinformationen ein.
2. Wähle auf der Startseite den Bereich Profil bearbeiten (obere rechte Ecke) und dann Instanzpasswort verwalten.
3. Rufen Sie die Anmeldedaten wie Benutzername, Passwort und Instanz-URL ab.

> **Note**
>
> <username>Wenn MFA für das Konto aktiviert ist, hängen Sie das MFA-Token an das Ende des Benutzerkennworts in der Basisauthentifizierung an: <password><MFA Token>

Weitere Informationen finden Sie in der Dokumentation unter Anwendungen erstellen. ServiceNow

OAuth 2.0-Anmeldeinformationen werden erstellt

Um OAuth2 2.0 im ServiceNow Connector zu verwenden, müssen Sie einen eingehenden Client () erstellen, um die Client-ID und den geheimen Clientschlüssel zu generieren:

1. Geben Sie auf der ServiceNow Anmeldeseite Ihre Kontoanmeldeinformationen ein.
2. Wählen Sie auf der Startseite Start Building aus.
3. Suchen Sie auf der App Engine Studio-Seite nach Anwendungsregistrierung.
4. Wählen Sie oben rechts „Neu“ aus.
5. Wählen Sie die Option Einen OAuth API-Endpunkt für externe Clients erstellen.
6. Nehmen Sie alle erforderlichen Änderungen an der OAuth Konfiguration vor und wählen Sie Aktualisieren.

   Beispiel für die Weiterleitungs-URL: https://us-east-1.console.aws.amazon.com/gluestudio/ oauth
7. Wählen Sie die neu erstellte OAuth Client-App aus, um die Client-ID und das Client-Geheimnis abzurufen.
8. Speichern Sie die Client-ID und das Client-Geheimnis für die weitere Verarbeitung.

So konfigurieren Sie die Konfiguration OAuth in einem Entwickler-Konto, das nicht für die Produktion bestimmt ist:

1. Erstellen Sie mithilfe des Themas Authentifizierungsprofil erstellen in der ServiceNow Dokumentation ein Authentifizierungsprofil.
2. Wählen Sie im Authentifizierungsprofil für OAuth die Option Typ als aus OAuth und wählen Sie den oben erstellten Inbound-Client aus, um die OAuth Entität festzulegen.
3. Wenn es mehrere Clients gibt, müssen Sie mehrere Authentifizierungsprofile erstellen, um die erforderliche OAuth Entität im Authentifizierungsprofil festzulegen.
4. Falls nicht konfiguriert, erstellen Sie eine REST-API-Zugriffsrichtlinie, um Zugriff auf die TABLE-API zu gewähren. Siehe REST-API-Zugriffsrichtlinie erstellen.

ServiceNow Verbindungen konfigurieren

Die Art der Gewährung bestimmt, wie mit Ihren Daten AWS Glue kommuniziert wird ServiceNow , um Zugriff auf diese anzufordern. Ihre Wahl wirkt sich auf die Anforderungen aus, die Sie erfüllen müssen, bevor Sie die Verbindung herstellen. ServiceNow unterstützt nur den Grant-Typ AUTHORIZATION_CODE für 2.0. OAuth

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf angewiesen ist, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer ServiceNow dorthin weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine ServiceNow Instanz gewähren muss.
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in zu erstellen ServiceNow und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin weitergeleitet, ServiceNow um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mit dem Aktualisierungstoken automatisch ohne Benutzerinteraktion aktualisiert werden.
- Öffentliche ServiceNow Dokumentation zur Erstellung einer verbundenen App für den OAuth Autorisierungscodefluss finden Sie unter Einrichten OAuth.

So konfigurieren Sie eine ServiceNow Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die Standardauthentifizierung sollte das Secret die verbundene App Consumer Secret mit `USERNAME` und `PASSWORD` als Schlüssel enthalten.
   b. Bei einem Typ zur Gewährung eines Autorisierungscodes sollte das Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   c. Hinweis: Sie müssen für jede Verbindung in ein Secret erstellen AWS Glue.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps ServiceNow.
   b. Geben Sie die INSTANCE_URL der ServiceNow Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden AWS Glue Aktionen übernehmen kann und über Berechtigungen verfügt:

```
{
```

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie den Authentifizierungstyp aus, den Sie für diese Verbindung verwenden möchten. AWS Glue

   i. Basic Auth: Dieser Authentifizierungstyp eignet sich gut für Automatisierungsszenarien, da er es ermöglicht, Benutzername und Passwort im Voraus mit den Berechtigungen eines bestimmten Benutzers in der ServiceNow Instanz zu verwenden. AWS Glue ist in der Lage, den Benutzernamen und das Passwort zur Authentifizierung zu verwenden. ServiceNow APIs Geben Sie die folgenden Eingaben nur bei Basic Auth ein: `Username` und. `Password`

   ii. OAuth2: Geben Sie die folgenden Eingaben nur bei OAuth2:`ClientId`,, `ClientSecretAuthorization URL`, `Authorization Token URL` ein.

e. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

f. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus ServiceNow Entitäten lesen

Voraussetzung

Ein ServiceNow Tables-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie pa_bucket oder incident.

Beispiel:

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2"
        "instanceUrl": "https://<instance-name>.service-now.com"
    }
)
```

ServiceNow Entitäts- und Felddetails:

ServiceNow Stellt für die folgenden Entitäten Endpunkte zum dynamischen Abrufen von Metadaten bereit, sodass die Operatorunterstützung für jede Entität auf Datentypebene erfasst wird.

Entität	Datentyp	Unterstützte Operatoren
Tabellen (dynamische Entitäten)	Ganzzahl	=,! =, <, <=, >, >=, ZWISCHEN
	BigDecimal	=,! =, <, <=, >, >=, ZWISCHEN
	Gleitkommazahl	=,! =, <, <=, >, >=, ZWISCHEN
	Long	=,! =, <, <=, >, >=, ZWISCHEN
	Datum	=,! =, <, <=, >, >=, ZWISCHEN
	DateTime	=,! =, <, <=, >, >=, ZWISCHEN
	Boolesch	=, !=
	String	=,! =, <, <=, >, >=, ZWISCHEN, WIE
	Struct	N/A

> **Note**
>
> Der Datentyp Struct wird in der Antwort des Konnektors in einen String-Datentyp umgewandelt.

> **Note**
>
> `DML_STATUS`ist ein zusätzliches benutzerdefiniertes Attribut, das für die Nachverfolgung von ERSTELLTEN/AKTUALISIERTEN Datensätzen verwendet wird.

Abfragen partitionieren

Partitionierung auf Feldbasis:

Sie können die zusätzlichen Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`, und angeben`UPPER_BOUND`, `NUM_PARTITIONS` ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Entity-Name	Partitionierung von Feldern	Datentyp
Dynamische Entität	sys_mod_count	Ganzzahl
	sys_created_on, sys_updated_on	DateTime

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in SPark SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

  ```
  "2024-01-30T06:47:51.000Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

In der folgenden Tabelle werden die Informationen zur Unterstützung von Feldern zur Partitionierung von Entitäten beschrieben:

Beispiel:

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2",
        "instanceUrl": "https://<instance-name>.service-now.com"
        "PARTITION_FIELD": "sys_created_on"
        "LOWER_BOUND": "2024-01-30T06:47:51.000Z"
        "UPPER_BOUND": "2024-06-30T06:47:51.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Datensatzbasierte Partitionierung:

Sie können die zusätzliche Spark-Option bereitstellen, NUM_PARTITIONS wenn Sie Parallelität in Spark nutzen möchten. Mit diesem Parameter wird die ursprüngliche Abfrage in eine NUM_PARTITIONS Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Bei der datensatzbasierten Partitionierung wird die Gesamtzahl der vorhandenen Datensätze von der ServiceNow API abgefragt und durch die angegebene Anzahl dividiert. NUM_PARTITIONS Die resultierende Anzahl von Datensätzen wird dann gleichzeitig von jeder Unterabfrage abgerufen.

- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
```

```
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2",
        "instanceUrl": "https://<instance-name>.service-now.com"
        "NUM_PARTITIONS": "2"
    }
```

ServiceNow Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für ServiceNow:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in ServiceNow.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. ServiceNow Rest-API-Version, die Sie verwenden möchten. Zum Beispiel: v1, v2, v3, v4.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes. Zum Beispiel: 2024-01-30T 06:47:51.000 Z.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes. Zum Beispiel: 2024-06-30T 06:47:51.000 Z.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen. Zum Beispiel: 10.
- `INSTANCE_URL`(String) — (Erforderlich) Eine gültige ServiceNow Instanz-URL im Format https://<instance-name>.service-now.com.
- `PAGE_SIZE`(Integer) — Definiert die Seitengröße für die Abfrage der Datensätze. Die Standard-Seitengröße ist 1.000. Wenn eine Seitengröße angegeben ist, wird nur die definierte Anzahl von Datensätzen pro API-Aufruf ServiceNow zurückgegeben, nicht der gesamte Datensatz. Der Connector stellt weiterhin die Gesamtzahl der Datensätze bereit und verarbeitet die Seitennummerierung unter Verwendung der angegebenen Seitengröße. Wenn Sie eine

größere Seitengröße benötigen, können Sie einen beliebigen Wert bis zu 10.000 wählen, was der maximal zulässige Wert ist. Jede angegebene Seitengröße, die 10.000 überschreitet, wird ignoriert. Stattdessen verwendet das System die maximal zulässige Seitengröße. Sie können die Seitengröße in der AWS Glue Studio-Benutzeroberfläche angeben, indem Sie eine Verbindungsoption `PAGE_SIZE` mit Ihrem gewünschten Wert hinzufügen. Zum Beispiel: 5000.

Einschränkungen und Hinweise für ServiceNow Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den ServiceNow Konnektor:

- Gemäß der SaaS-Dokumentation `sys_mod_count` sind`sys_created_on`,`sys_updated_on`, und vom System generierte Felder. Der Connector verwendet SaaS APIs , um diese Felder im Antworttext bereitzustellen.
  - Wenn SaaS diese Felder für keine Entität generiert, kann die filterbasierte Partitionierung nicht unterstützt werden.
- Wenn SaaS APIs keine `sys_updated_on` Felder in der Antwort zurückgibt`sys_created_on`, `DML_STATUS` kann dies nicht berechnet werden.

Verbindung zu Slack herstellen in AWS Glue Studio

Slack ist eine Kommunikations-App für Unternehmen, mit der Benutzer Nachrichten und Anhänge über verschiedene öffentliche und private Kanäle senden können. Wenn du ein Slack-Nutzer bist, kannst du dich mit deinem Slack-Account AWS Glue verbinden. Anschließend kannst du Slack als Datenquelle in deinen ETL-Jobs verwenden. Führe diese Jobs aus, um Daten zwischen Slack und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Slack
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Slack konfigurieren
- Slack-Verbindungen konfigurieren
- Aus Slack-Entitäten lesen
- Verbindungsoptionen für Slack
- Einschränkungen
- Einen neuen Slack-Account erstellen und die Client-App konfigurieren

AWS Glue Unterstützung für Slack

AWS Glue unterstützt Slack wie folgt:

Als Quelle unterstützt?

Ja. Du kannst AWS Glue ETL-Jobs verwenden, um Daten von Slack abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Slack-API-Versionen

Slack API v2.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM und Amazon EC2. Amazon CloudWatch Logs Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Slack konfigurieren

Bevor du Daten AWS Glue zu oder von Slack übertragen kannst, musst du die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Du musst ein Slack-Konto haben. Weitere Informationen finden Sie unter Einen neuen Slack-Account erstellen und die Client-App konfigurieren.

Wenn du diese Anforderungen erfüllst, bist du bereit, dich mit deinem Slack-Account AWS Glue zu verbinden.

Slack-Verbindungen konfigurieren

Slack unterstützt den `AUTHORIZATION_CODE` Grant-Typ für 2. OAuth

Diese Art der Gewährung wird als „dreibeiniges“ angesehen, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Slack weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Slack-Instanz gewähren muss.

Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Slack zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über

die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Slack weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Grant-Typ führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken läuft nach einer Stunde nach der Erstellung ab. Ein neues Zugriffstoken kann mit dem Aktualisierungstoken abgerufen werden.

Weitere Informationen zum Erstellen einer verbundenen App für den OAuth Autorisierungscode-Flow findest du unter Slack-API.

So konfigurierst du eine Slack-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details. Sie müssen ein Geheimnis für die Verbindung in erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Slack aus.
   b. Stelle die Slack-Umgebung bereit.
   c. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Slack-Entitäten lesen

Voraussetzungen

- Ein Slack-Objekt, aus dem du lesen möchtest.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Unterhaltungen	Ja	Ja	Nein	Ja	Ja

Beispiel

```
slack_read = glueContext.create_dynamic_frame.from_options(
    connection_type="slack",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversations/C058W38R5J8"
    }
)
```

Entitäts- und Felddetails in Slack

Entität	Feld	Datentyp	Unterstützte Operatoren
Unterhaltungen	Anhänge	Auflisten	N/A
Unterhaltungen	bot_id	String	N/A
Unterhaltungen	blocks	Auflisten	N/A
Unterhaltungen	client_msg_id	String	N/A
Unterhaltungen	ist mit einem Stern markiert	Boolesch	N/A
Unterhaltungen	zuletzt gelesen	String	N/A
Unterhaltungen	neueste_Antwort	String	N/A
Unterhaltungen	Reaktionen	Auflisten	N/A
Unterhaltungen	Antworten	Auflisten	N/A
Unterhaltungen	reply_count	Ganzzahl	N/A
Unterhaltungen	reply_users	Auflisten	N/A
Unterhaltungen	reply_users_count	Ganzzahl	N/A
Unterhaltungen	angemeldet	Boolesch	N/A
Unterhaltungen	Untergruppe	String	N/A
Unterhaltungen	text	String	N/A
Unterhaltungen	Mannschaft	String	N/A
Unterhaltungen	thread_ts	String	N/A
Unterhaltungen	ts	String	GLEICH_ALS, ZWISCHEN, KLEINER_A

Entität	Feld	Datentyp	Unterstützte Operatoren
			LS, KLEINER_A LS_ODER_G LEICH_ZU, GRÖSSER_A LS, GRÖSSER_A LS_ODER_G LEICH_ZU
Unterhaltungen	Typ	String	N/A
Unterhaltungen	user	String	N/A
Unterhaltungen	einladen	String	N/A
Unterhaltungen	Root	Struct	N/A
Unterhaltungen	ist_gesperrt	Boolesch	N/A
Unterhaltungen	files	Auflisten	N/A
Unterhaltungen	Raum	Struct	N/A
Unterhaltungen	Hochladen	Boolesch	N/A
Unterhaltungen	anzeigen_als_bot	Boolesch	N/A
Unterhaltungen	channel	String	N/A
Unterhaltungen	no_notifications	Boolesch	N/A
Unterhaltungen	Permalink	String	N/A
Unterhaltungen	angeheftet an	Auflisten	N/A
Unterhaltungen	gepinned_info	Struct	N/A
Unterhaltungen	bearbeitet	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Unterhaltungen	app_id	String	N/A
Unterhaltungen	bot_profile	Struct	N/A
Unterhaltungen	Metadaten	Struct	N/A

Abfragen partitionieren

Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für einen gültigen Wert:`"2024-07-01T00:00:00.000Z"`.
- `UPPER_BOUND`: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt.

Entity Name	Feld partitionieren	Datentyp
Unterhaltungen	ts	String

Beispiel

```
slack_read = glueContext.create_dynamic_frame.from_options(
    connection_type="slack",
```

```
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversations/C058W38R5J8",
        "PARTITION_FIELD": "ts"
        "LOWER_BOUND": "2022-12-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-09-23T15:00:00.000Z"
        "NUM_PARTITIONS": "2"
    }
)
```

Verbindungsoptionen für Slack

Im Folgenden sind die Verbindungsoptionen für Slack aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Name der unterstützten Entität. Beispiel: `conversations/C058W38R5J8`.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Felder, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `NUM_PARTITIONS`(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der zu lesenden Partitionen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Slack-Connector:

- Die datensatzbasierte Partitionierung wird nicht unterstützt, da der Connector keine Möglichkeit bietet, die Gesamtzahl der in einer bestimmten Konversation verfügbaren Datensätze (Nachrichten) abzurufen.

Einen neuen Slack-Account erstellen und die Client-App konfigurieren

Ein Slack-Konto erstellen

1. Öffne die Slack-Startseite, um dich für ein Konto zu registrieren.

2. Wähle MIT E-MAIL-ADRESSE REGISTRIEREN. Geben Sie Ihre E-Mail-ID ein und wählen Sie Weiter.
3. Gib den sechsstelligen Code ein, der an deine E-Mail-Adresse gesendet wurde. Er leitet dich weiter, ob du einen Workspace erstellen oder einem bestehenden Workspace beitreten möchtest.
4. Wähle Workspace erstellen, um einen neuen Workspace zu erstellen. Im Rahmen des Einrichtungsprozesses wirst du aufgefordert, einige Fragen zu beantworten.
   - Name der Firma
   - Ihr Name
   - Um Kollegen per E-Mail hinzuzufügen
   - Woran arbeitet Ihr Team? (Dies wird der Kanalname sein)
5. Füllen Sie die Eingabefelder für diese Fragen aus und fahren Sie fort. Ihr Konto kann jetzt verwendet werden.

Eine Slack-Entwickler-App erstellen

1. Melde dich bei deinem Slack-Account an und melde dich in deinem Slack-Workspace an.
2. Wähle im Workspace-Menü Tools und Einstellungen und dann Apps verwalten aus.
3. Wähle im App-Verzeichnismenü von Slack die Option Build aus.
4. Wähle auf der Seite „Deine Apps“ die Option App erstellen aus.
5. Wählen Sie auf der Seite App erstellen die Option Von Grund auf neu aus.
6. Fügen Sie im sich öffnenden Dialogfeld App benennen und Workspace auswählen einen App-Namen hinzu und wählen Sie einen Workspace aus, in dem Ihre App bereitgestellt werden soll. Wähle dann App erstellen aus.
7. Notieren Sie sich Ihre Client-ID und Ihr Secret, die in den App-Anmeldeinformationen angezeigt werden
8. Gehen Sie in der Seitenleiste OAuth und Berechtigungen zu Bereichen und wählen Sie Bereich hinzufügen aus. OAuth Du kannst die Weiterleitung URLs zur Konfiguration zu deiner App hinzufügen, um automatisch die Schaltfläche „Zu Slack hinzufügen“ zu generieren oder deine App zu verteilen. Scrolle zum URLs Abschnitt „Weiterleitung“ und wähle „Neue Weiterleitungs-URL hinzufügen“ und speichere.

9. Scrollen Sie dann zum Abschnitt OAuth Tokens für Ihren Workspace und wählen Sie In Workspace installieren aus.
10. Wähle in dem sich öffnenden Dialogfeld, das dich darüber informiert, dass die von dir erstellte App um Erlaubnis bittet, auf den Slack-Workspace zuzugreifen, mit dem du sie verbinden möchtest, und wähle Zulassen aus.
11. Nach erfolgreichem Abschluss zeigt die Konsole den Bildschirm „ OAuth Tokens für deinen Workspace“ an.
12. Kopieren und speichern Sie auf dem Bildschirm „ OAuth Tokens für Ihren Workspace“ das OAuth Token, mit dem Sie sich verbinden möchten AWS Glue
13. Als Nächstes rufst du deine Slack-Team-ID ab. Wähle im Workspace-Menü von Slack Tools und Einstellungen und dann Apps verwalten aus. Du findest deine Team-ID in der URL der Seite, die geöffnet wird.
14. Um deine App öffentlich zu verteilen, kannst du sie aktivieren, indem du in der Seitenleiste auf die Schaltfläche Vertrieb verwalten gehst. Scrollen Sie nach unten zum Abschnitt „Deine App mit anderen Workspaces teilen“ und wähle „Hardcodierte Informationen entfernen“. Erteilen Sie Ihre Zustimmung und wählen Sie Active Public Distribution.
15. Ihre App wird jetzt öffentlich vertrieben. Um auf die Entität zuzugreifen APIs, muss die App zu jedem Workspace-Channel hinzugefügt werden, von dem aus der Benutzer darauf zugreifen möchte.
16. Melde dich in deinem Slack-Account an und öffne den Workspace, auf dessen Channel zugegriffen werden muss.
17. Öffne im Workspace den Channel, auf den die App zugreifen möchte, und wähle den Kanaltitel aus. Wähle im Pop-up den Tab Integrationen aus und füge die App hinzu. Auf diese Weise wird die App in den Kanal integriert, um Zugriff auf dessen API zu erhalten.

    Die OAuth 2.0-Client-ID muss über eine oder mehrere autorisierte Weiterleitungen verfügen URLs. Weiterleitungen URLs haben das folgende Format:

    > **Note**
    >
    > Appflow-Weiterleitungen können URLs sich ändern, nachdem die Weiterleitung URLs für die AWS Glue Plattform verfügbar ist. Client ID und Client Secret stammen aus den Einstellungen für Ihre OAuth 2.0-Client-ID.

Die Weiterleitungs-URL kann eine der folgenden sein	
Umleitungs-URL für die Gamma-Umgebung	
https://us-west-2.console.aws.amazon.com/appflow/oauth	https://us-east-1.awsc-integ.aws.amazon.com/appflow/oauth
https://us-east-2.console.aws.amazon.com/appflow/oauth	
https://us-west-1.console.aws.amazon.com/appflow/oauth	
https://ap-south-1.console.aws.amazon.com/appflow/oauth	
https://ap-southeast-1.console.aws.amazon.com/appflow/oauth	
https://ap-southeast-2.console.aws.amazon.com/appflow/oauth	
https://ap-northeast-1.console.aws.amazon.com/appflow/oauth	
https://ap-northeast-2.console.aws.amazon.com/appflow/oauth	
https://ca-central-1.console.aws.amazon.com/appflow/oauth	
https://eu-central-1.console.aws.amazon.com/appflow/oauth	
https://eu.west-1.console.aws.amazon.com/appflow/oauth	

Die Weiterleitungs-URL kann eine der folgenden sein	
Umleitungs-URL für die Gamma-Umgebung	
https://eu-west-2.console.aws.amazon.com/appflow/oauth	
https://eu-west-3.console.aws.amazon.com/appflow/oauth	
https://sa-east-1.console.aws.amazon.com/appflow/oauth	
https://us-west-2.awsc-integ.aws.amazon.com/appflow/oauth	
https://af-south-1.console.aws.amazon.com/appflow/oauth	

Verbindung zu Smartsheet herstellen

Smartsheet ist ein SaaS-Produkt für Arbeitsmanagement und Zusammenarbeit. Grundsätzlich ermöglicht Smartsheet Benutzern, tabellenähnliche Objekte zu verwenden, um Geschäftsdaten zu erstellen, zu speichern und zu nutzen.

Themen

- AWS Glue Unterstützung für Smartsheet
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Smartsheet konfigurieren
- Smartsheet-Verbindungen konfigurieren
- Aus Smartsheet-Entitäten lesen
- Smartsheet-Verbindungsoptionen
- Ein Smartsheet-Konto erstellen
- Einschränkungen

AWS Glue Unterstützung für Smartsheet

AWS Glue unterstützt Smartsheet wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Smartsheet abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Smartsheet-API-Versionen

v2.0

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

In der folgenden Beispielrichtlinie werden die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen beschrieben. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole — Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess — Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Smartsheet konfigurieren

Bevor Sie die Übertragung von Smartsheet verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie haben ein Smartsheet-Konto mit E-Mail und Passwort. Weitere Informationen zum Erstellen eines Kontos finden Sie unter Ein Smartsheet-Konto erstellen.
- Ihr Smartsheet-Konto hat API-Zugriff mit einer gültigen Lizenz.
- Ihr Smartsheet-Konto verfügt über einen Pro-Preisplan für `Sheets` Entität und einen Enterprise-Preisplan mit Event Reporting-Add-On für `Events` Entität.

Wenn Sie diese Anforderungen erfüllen, sind Sie bereit, eine Verbindung AWS Glue zu Ihrem Smartsheet-Konto herzustellen. Für typische Verbindungen müssen Sie in Smartsheet nichts weiter tun.

Smartsheet-Verbindungen konfigurieren

Smartsheet unterstützt den `AUTHORIZATION_CODE` Zuschusstyp für. OAuth2

Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, `OAuth` da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet

werden. Benutzer können sich dafür entscheiden, ihre eigene verbundene App in Smartsheet zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Smartsheet weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.

Eine öffentliche Smartsheet-Dokumentation zur Erstellung einer verbundenen App für den AUTHORIZATION_CODE-Flow OAuth finden Sie unter Smartsheet. APIs

So konfigurieren Sie eine Smartsheet-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:

   Für vom Kunden verwaltete verbundene App — Secret sollte die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

   > **Note**
   >
   > Es ist ein Muss, pro Verbindung ein Geheimnis zu erstellen AWS Glue.

2. Stellen Sie im unter Datenverbindungen eine Verbindung her AWS Glue Studio, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Smartsheet aus.
   b. Geben Sie das Smartsheet `instanceUrl` an, mit dem Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die IAM-Rolle aus, für die AWS Glue Sie die folgenden Aktionen übernehmen können und über Berechtigungen verfügen:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
```

```
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, die Sie für diese Verbindung verwenden möchten AWS Glue , um die Token einzufügen.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Smartsheet-Entitäten lesen

Voraussetzungen

Ein `Smartsheet` Objekt, aus dem Sie lesen möchten. In der Tabelle mit den unterstützten Entitäten unten können Sie die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Listenblatt	Ja	Ja	Nein	Ja	Nein
Zeilen-Metadaten	Ja	Ja	Nein	Ja	Nein
Blattmetadaten	Nein	Nein	Nein	Ja	Nein
Blattdaten	Ja	Ja	Ja	Ja	Nein
Ereignis	Ja	Ja	Nein	Ja	Nein

Beispiel

```
Smartsheet_read = glueContext.create_dynamic_frame.from_options(
    connection_type="smartsheet",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "list-sheets",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://api.smartsheet.com"
    })
```

Entitäts- und Felddetails in Smartsheet

Entität	Feld	Datentyp	Unterstützte Betreiber
Blätter auflisten	id	Long	N/A
	Zugriffsebene	String	N/A
	createdAt	DateTime	N/A
	Geändert auf	DateTime	N/A
	Name	String	N/A
	Permalink	String	N/A
	Geändert seit	DateTime	>=
	version	Ganzzahl	N/A
	Quelle	Struct	N/A
Zeilen-Metadaten	id	Long	N/A
	Blatt-ID	Long	N/A
	Zugriffsebene	String	N/A
	Anhänge	Auflisten	N/A
	Spalten	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Betreiber
	Bedingtes Format	String	N/A
	createdAt	DateTime	N/A
	Erstellt von	Struct	N/A
	Diskussionen	Auflisten	N/A
	Beweise	Struct	N/A
	erweitert	Boolesch	N/A
	Herausgefiltert	Boolesch	N/A
	Format	String	N/A
	inCriticalPath	Boolesch	N/A
	verschlossen	Boolesch	N/A
	lockedForUser	Boolesch	N/A
	Geändert am	DateTime	N/A
	Geändert von	Struct	N/A
	Permalink	String	N/A
	Zeilennummer	Ganzzahl	N/A
	version	Ganzzahl	N/A
	totalRowCount	Ganzzahl	N/A
	rowsModifiedSince	DateTime	>
	Filter-ID	Long	“="
	Geschwister-ID	Long	N/A

Entität	Feld	Datentyp	Unterstützte Betreiber
	ID der Eltern	Long	N/A
Metadaten des Blattes	id	Long	N/A
	Von ID	Long	N/A
	ownerId	Long	N/A
	Zugriffsebene	String	N/A
	Anhänge	Auflisten	N/A
	Spalten	Auflisten	N/A
	createdAt	DateTime	N/A
	crossSheetReferences	Auflisten	N/A
	Abhängigkeiten aktiviert	Boolesch	N/A
	Diskussionen	Auflisten	N/A
	effectiveAttachmentOptions	Auflisten	N/A
	Favorit	Boolesch	N/A
	Gantt aktiviert	Boolesch	N/A
	hasSummaryFields	Boolesch	N/A
	Geändert am	DateTime	N/A
	Name	String	N/A
	owner	String	N/A
	Permalink	String	N/A

Entität	Feld	Datentyp	Unterstützte Betreiber
	Einstellungen des Projekts	Struct	N/A
	readOnly	Boolesch	N/A
	resourceManagementEnabled	Boolesch	N/A
	showParentRowsForFilters	Boolesch	N/A
	Quelle	Struct	N/A
	Übersicht	Struct	N/A
	totalRowCount	Ganzzahl	N/A
	Benutzerberechtigungen	Struct	N/A
	Benutzereinstellungen	Struct	N/A
	version	Ganzzahl	N/A
	Workspace	Struct	N/A
	-Filter	Auflisten	N/A
	Gantt-Konfiguration	Struct	N/A
	resourceManagementType	String	N/A
	cellImageUploadAktiviert	Boolesch	N/A
	isMultiPicklistAktiviert	Boolesch	N/A
--Ereignisse	eventId	String	N/A

Entität	Feld	Datentyp	Unterstützte Betreiber
	objectType	String	N/A
	action	String	N/A
	objectId	Long	N/A
	Zeitstempel des Ereignisses	DateTime	N/A
	userId	Long	N/A
	requestUserId	Long	N/A
	accessTokenName	String	N/A
	Quelle	String	N/A
	Zusätzliche Einzelheiten	Struct	N/A
	since	DateTime	>=

Entitäten mit dynamischen Metadaten:

Für die folgende Entität bietet Smartsheet einen Endpunkt zum dynamischen Abrufen von Metadaten, sodass die Bedienerunterstützung auf Datentypebene erfasst werden kann.

Entität	Datentyp	Unterstützte Operatoren
Blattdaten	String	N/A
Blattdaten	Long	"="
Blattdaten	Ganzzahl	N/A
Blattdaten	DateTime	>

Beispiel

```
Smartsheet_read = glueContext.create_dynamic_frame.from_options(
    connection_type="smartsheet",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "list-sheets",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://api.smartsheet.com"
    }
```

Smartsheet-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Smartsheet aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Smartsheet.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Die Smartsheet-Rest-API-Version, die Sie verwenden möchten. Zum Beispiel: v2.0.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. URL der Smartsheet-Instanz.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Ein Smartsheet-Konto erstellen

1. Melden Sie sich für ein Smartsheet-Konto an, indem Sie auf die Smartsheet-Anmeldeseite zugreifen.
2. Wählen Sie Create one, um ein neues Konto zu erstellen, oder melden Sie sich mit Ihrem registrierten Google-, Microsoft- oder Apple-Konto an.
3.
4.
5. Öffnen Sie die Bestätigungs-E-Mail von Smartsheet und wählen Sie den Bestätigungslink, um Ihr Konto zu verifizieren.

Sie werden standardmäßig den Testplan abonnieren.

6. Wählen Sie in der unteren linken Ecke das Kontosymbol und dann Lizenzen hinzufügen/Upgrade hinzufügen, um Ihren Preisplan zu aktualisieren.

> **Note**
>
> Dies ist für den Zugriff auf Event Reporting erforderlich, das ein Add-on im Enterprise-Plan ist.

7. Wählen Sie im Enterprise-Plan die Option Kontaktieren Sie uns aus, um beim Support-Team ein Account-Upgrade anzufordern.
8. Geben Sie im Formular für Support-Anfragen die erforderlichen Details und Ihre Anforderungen für ein Upgrade des Plans an.

   Damit ist das Upgrade auf den Enterprise-Plan abgeschlossen.

`OAuth2.0`Anmeldeinformationen werden erstellt

1. Nachdem Sie den Preisplan Ihres Kontos aktualisiert haben, um Zugriff auf die Entwicklertools zu erhalten, greifen Sie auf Smartsheet-Entwickler zu.

   Sie erhalten eine Aktivierungs-E-Mail.
2. Öffnen Sie eine Aktivierungs-E-Mail von Smartsheet und wählen Sie den Aktivierungslink, um die Entwicklertools in Ihrem Konto zu aktivieren.

   Mit dem Entwicklertool können Sie die App erstellen.
3. Öffnen Sie die Startseite Ihres Smartsheet-Kontos und wählen Sie Konto aus, um den Zugriff zu überprüfen.
4. Wählen Sie Developer Tools aus der Liste der Dienste aus und geben Sie die Details des Entwicklerprofils ein.
5. Wählen Sie „Neue App erstellen“.
6. Geben Sie die folgenden Details in das App-Registrierungsformular ein:

   - Name — Name der App.
   - Beschreibung — Beschreibung der App.
   - URL — URL, mit der Sie Ihre App starten können, oder die URL der Landingpage.

- Kontakt/Support — Kontaktinformationen für das Support-Team.
- Weiterleitungs-URL — URL (auch als Rückruf-URL bezeichnet) innerhalb Ihrer Anwendung, die die 2.0-Anmeldeinformationen erhält. OAuth

7. Wählen Sie Save (Speichern) aus.

   Smartsheet weist Ihrer App eine Client-ID und einen geheimen Client-Schlüssel zu. Notieren Sie sich diese Werte für die nächsten Schritte. Sie können sie auch später im Abschnitt Entwicklertools erneut nachschlagen.

Einschränkungen

Smartsheet unterstützt keine feldbasierte oder datensatzbasierte Partitionierung.

Verbindung zu Snapchat-Anzeigen herstellen in AWS Glue Studio

Snapchat ist eine Multimedia-Instant-Messaging-Anwendung und ein Dienst, die von Snap Inc., ursprünglich Snapchat Inc., entwickelt wurden. Eines der Hauptmerkmale von Snapchat ist, dass Bilder und Nachrichten normalerweise nur für kurze Zeit verfügbar sind, bevor sie für ihre Empfänger nicht mehr zugänglich sind. Snapchat-Marketing sind Beiträge, für die Unternehmen zahlen können, um sie Snapchat-Nutzern zur Verfügung zu stellen.

Themen

- AWS Glue Unterstützung für Snapchat-Anzeigen
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Snapchat-Anzeigen
- Konfiguration von Snapchat Ads-Verbindungen
- Aus Snapchat Ads-Entitäten lesen
- Verbindungsoptionen für Snapchat Ads
- Erstellen eines Snapchat-Werbekontos und Konfiguration der Client-App
- App in deinem Snapchat Ads-Konto erstellen

AWS Glue Unterstützung für Snapchat-Anzeigen

AWS Glue unterstützt Snapchat-Anzeigen wie folgt:

Wird es als Quelle unterstützt?

Ja. Du kannst AWS Glue ETL-Jobs verwenden, um Daten aus Snapchat Ads abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen von Snapchat Ads

v1.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM und Amazon EC2. Amazon CloudWatch Logs Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über

die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Snapchat-Anzeigen

Bevor du die Übertragung von Snapchat Ads nutzen AWS Glue kannst, musst du die folgenden Voraussetzungen erfüllen:

Mindestanforderungen

- Du hast ein Snapchat Ads-Konto. Weitere Informationen zum Erstellen eines Kontos findest du unter Erstellen eines Snapchat-Werbekontos und Konfiguration der Client-App .
- Du hast in deinem Snapchat Ads-Konto eine OAuth2 App erstellt. Diese Integration stellt die Anmeldeinformationen bereit, die für den sicheren Zugriff auf deine Daten AWS Glue verwendet werden, wenn authentifizierte Anrufe an dein Konto getätigt werden. Weitere Informationen finden Sie unter App in deinem Snapchat Ads-Konto erstellen .

Wenn du diese Anforderungen erfüllst, kannst du dich mit deinem Snapchat Ads-Konto verbinden AWS Glue . In Snapchat Ads ist eine verbundene App ein Framework, das externen Anwendungen beispielsweise AWS Glue den Zugriff auf deine Snapchat Ads-Daten ermöglicht.

Konfiguration von Snapchat Ads-Verbindungen

Snapchat Ads unterstützt nur den `AUTHORIZATION_CODE` Zuschusstyp.

Diese Art der Gewährung gilt als „dreibeiniges“, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an den Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App (AWS Glue verwaltete Client-Anwendung) verlassen, für die er außer der URL seiner Snapchat Ads-Instanz keine OAuth zugehörigen Informationen angeben muss. Die AWS Glue Konsole leitet den Nutzer zu Snapchat Ads weiter, wo er sich anmelden und die angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Snapchat Ads-Instanz gewähren muss.

Nutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Snapchat Ads zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Snapchat Ads weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.

Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken läuft nach 1 Stunde nach der Erstellung ab. Ein neues Zugriffstoken kann mit dem Aktualisierungstoken abgerufen werden.

Weitere Informationen zum Erstellen einer verbundenen App für den OAuth Autorisierungscode-Flow findest du unter Ads-API.

So konfigurierst du eine Snapchat Ads-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Snapchat Ads aus.
   b. Stellen Sie die Snapchat Ads-Umgebung bereit.
   c. Wählen Sie die IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Snapchat Ads-Entitäten lesen

Voraussetzungen

- Ein Snapchat Ads-Objekt, aus dem du lesen möchtest. In der Tabelle mit den unterstützten Entitäten unten kannst du die verfügbaren Entitäten überprüfen.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Organisation	Nein	Nein	Nein	Ja	Nein
Konto hinzufügen	Nein	Nein	Nein	Ja	Nein
Kreativ	Nein	Nein	Nein	Ja	Nein
Medien	Nein	Nein	Nein	Ja	Nein
Kampagne	Ja	Nein	Nein	Ja	Nein
Anzeige unter Werbekonto	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Anzeige unter Kampagne	Nein	Nein	Nein	Ja	Nein
Werbetrupp	Ja	Nein	Nein	Ja	Nein
Segment	Nein	Nein	Nein	Ja	Nein

Beispiel

```
snapchatads_read = glueContext.create_dynamic_frame.from_options(
    connection_type="snapchatAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "organization",
        "API_VERSION": "v1"
    }
)
```

Angaben zur Entität und zum Feld von Snapchat Ads

Snapchat Ads lädt die verfügbaren Felder dynamisch unter der ausgewählten Entität. Je nach Datentyp des Felds werden die folgenden Filteroperatoren unterstützt.

Datentyp des Felds	Unterstützte Filteroperatoren
Boolesch	=

Partitionierung von Abfragen

- Feldbasierte Partitionierung: Wird nicht unterstützt.
- Datensatzbasierte Partitionierung: Nicht unterstützt.

Verbindungsoptionen für Snapchat Ads

Im Folgenden sind die Verbindungsoptionen für Snapchat Ads aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name der Snapchat Ads-Entität. Beispiel: `campaign` .
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Rest-API-Version von Snapchat Ads, die Sie verwenden möchten. Der Wert wird v1 sein, da Snapchat Ads derzeit nur Version v1 unterstützt.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Durch Kommas getrennte Liste von Spalten, die Sie für die ausgewählte Entität auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Erstellen eines Snapchat-Werbekontos und Konfiguration der Client-App

Themen

- Melde dich für Snapchat Ads an
- Schritte zum Erstellen eines Snapchat-Werbekontos

Melde dich für Snapchat Ads an

So meldest du dich für Snapchat Ads an:

1. Navigiere zum Snapchat Ads Manager. Wähle Anmelden neben Neu bei Snapchat? .
2. Folgen Sie auf dem Bildschirm Konto erstellen den Anweisungen, um Ihren Firmennamen, Ihre E-Mail-Adresse, Ihr Passwort usw. einzugeben. Wählen Sie Weiter.
3. Geben Sie auf dem Bildschirm „Ihr Profil erstellen“ Werte für „Benutzername“ und „Website“ (optional) ein und wählen Sie „Konto erstellen“. Dadurch haben Sie die Möglichkeit, auf dem Bildschirm „Ihr Profil bearbeiten“ ein Profilfoto und eine Biografie hinzuzufügen. Wählen Sie Bestätigen aus.

4. Füllen Sie auf dem Bildschirm mit den Geschäftsinformationen die erforderlichen Felder wie Land, Währung, Telefonnummer, GSTIN usw. aus und schließen Sie die Kontoerstellung ab, indem Sie Weiter wählen.

Schritte zum Erstellen eines Snapchat-Werbekontos

So erstellst du ein Snapchat Ad-Konto:

1. Melde dich bei Ads Manager an. Klicke dann auf die Navigation in der oberen Ecke und wähle Werbekonten aus.
2. Wählen Sie + Neues Werbekonto. Geben Sie die Daten Ihres Werbetreibenden ein:

   - Wählen Sie aus, ob Sie eine Agentur sind, die Anzeigen im Namen eines Werbetreibenden kauft. Wenn Sie „Ja“ auswählen, wird Ihre Anzeige möglicherweise abgelehnt, wenn sie Ausrichtungsparameter wie Alter, Geschlecht oder PLZ verwendet. Die Ausrichtung auf das Mindestalter kann auf Personen bis zu 21 Jahren angewendet werden.
   - Wählen Sie aus, ob in Ihrem Werbekonto Wohnungsanzeigen, Kreditanzeigen oder Stellenausschreibungen geschaltet werden sollen. Wenn Sie „Ja“ auswählen, wird Ihre Anzeige möglicherweise abgelehnt, wenn sie Ausrichtungsparameter wie Alter, Geschlecht oder PLZ verwendet. Die Ausrichtung auf das Mindestalter kann auf Personen bis zu 21 Jahren angewendet werden.
   - Wählen Sie aus, ob Sie das Werbekonto für politische Werbung verwenden möchten. Wenn Sie eine politische Anzeige schalten, geben Sie die politische Organisation oder Interessenvertretung an, die die Anzeige finanziert. Wenn Sie die politische Organisation nicht korrekt angeben, werden Ihre Anzeigen möglicherweise abgelehnt. Sie müssen außerdem das obligatorische, verlinkte Formular zur Überprüfung politischer Anzeigen ausfüllen, bevor Sie Anzeigen einreichen können.
3. Wählen Sie Kontodetails und geben Sie Ihre Kontoinformationen für Anzeigen ein:

Feld	Beschreibung
Name	Name Ihres Werbekontos.
Kontotyp	Wählen Sie Ihren Kontotyp aus, falls er nicht automatisch ausgefüllt wird.

Feld	Beschreibung
Art der Abrechnung	Wenn Sie kontinuierlich Zugriff auf Ihre Finanzierungsquelle haben möchten, wählen Sie „Revolving“. Auf diese Weise kannst du deine Kreditlinie wieder auffüllen und diese Option wird häufig für die Schaltung von Anzeigen im Ads Manager ausgewähl t. Wenn du es vorziehst, eine einmalige Kampagne mit einer Ausgabenobergrenze zu starten, kannst du Insertion Order wählen. Wenn du ein Konto mit einem Insertion sauftrag erstellst, aber später eine Ausgabenobergrenze festlegen möchtest, kannst du die Ausgabenobergrenze bei 0$ belassen. Auf diese Weise gilt für das Konto ein Ausgabenlimit von 0 USD, Sie können es jedoch später anwenden.
Werbeorganisation	Die Organisation, die Werbung kauft.
Abrechnungszentrum	Wählen Sie das Abrechnungszentrum aus, für das Sie Rechnungen erhalten möchten. Sie können das Rechnungscenter verwenden, das automatisch erstellt wurde, als Sie Ihr Unternehmen zum ersten Mal in Anspruch genommen haben, oder ein anderes Rechnungscenter hinzufügen, indem Sie den Schritten im Business Help Center folgen.
Währung	Wählen Sie Ihre Währung.
Obergrenze ausgeben	Wenn Sie „Insertion Order“ als Abrechnun gsart ausgewählt haben, geben Sie ein Budget für das Werbekonto ein.

Feld	Beschreibung
Zeitzone	Wählen Sie Ihre Zeitzone aus.

4. Wählen Sie Konto erstellen. Dein Werbekonto wird erstellt und du findest es im Bereich „Werbekonten“ von Ads Manager. Um mit der Veröffentlichung von Anzeigen zu beginnen, müssen Sie eine Zahlungsmethode eingeben. Sie können Ihrem Werbekonto auch Mitglieder hinzufügen.
5. Wähle aus, ob du eine bestehende Zahlung verwenden oder eine neue erstellen möchtest. Wählen Sie dann Zahlungsmethode speichern.
6. Wähle alle Mitglieder aus, die du zu deinem Unternehmen eingeladen hast, um sie dem Werbekonto hinzuzufügen. Weitere Informationen zu den Rollen und Berechtigungen, die zugewiesen werden können, findest du unter Rollen und Berechtigungen im Überblick. Hinzugefügte Mitglieder können sich dann bei Ads Manager anmelden und auf dieses Werbekonto zugreifen. Wenn du fertig bist, speichere deine Mitglieder.

Weitere Informationen zu Werbekonten findest du unter https://businesshelp.snapchat.com/s/article/roles-permissions?language=en_US https://businesshelp.snapchat.com/s/ article/roles-permissions?language=de_DE

App in deinem Snapchat Ads-Konto erstellen

Um den Zugriff auf die Marketing-API von Snapchat zu aktivieren, stellen Sie sicher, dass Sie ein Geschäftskonto eingerichtet haben. Folgen Sie dann den nachstehenden Schritten.

1. Melde dich bei Ads Manager an. Wähle dann das Menü in der oberen linken Ecke und dann Business Dashboard und dann Unternehmensdetails aus.
2. Wählen Sie + OAuth App.
3. Geben Sie Ihren App-Namen ein und fügen Sie die folgende URL als Snap Redirect-URI hinzu`https://<aws-region>.console.aws.amazon.com/gluestudio/oauth`. Wenn Sie beispielsweise die Region us-west-1 verwenden, wäre die URL. `https://us-west-1.console.aws.amazon.com/gluestudio/oauth) and choose` **`Create OAuth App`** Wählen Sie „App erstellen OAuth “.
4. Ihre App-Anmeldeinformationen (Client-ID und Client Secret) werden angezeigt. Speichern Sie sie, da sie zum Herstellen einer Verbindung benötigt werden.

Verbindung zu Snowflake herstellen in AWS Glue Studio

> **Note**
>
> Sie können Folgendes verwenden … AWS Glue für Spark zum Lesen und Schreiben in Tabellen in Snowflake in AWS Glue 4.0 und spätere Versionen. Um eine Snowflake-Verbindung zu konfigurieren mit AWS Glue Jobs programmgesteuert, siehe. Redshift-Verbindungen

AWS Glue bietet integrierte Unterstützung für Snowflake. AWS Glue Studio bietet eine visuelle Oberfläche, über die Sie eine Verbindung zu Snowflake herstellen, Datenintegrationsjobs erstellen und diese auf dem ausführen können AWS Glue Studio serverlose Spark-Laufzeit.

AWS Glue Studio erstellt eine einheitliche Verbindung für Snowflake. Weitere Informationen finden Sie unter Überlegungen.

Themen

- Erstellen einer Snowflake-Verbindung
- Erstellen eines Snowflake-Quellknotens
- Erstellen eines Snowflake-Zielknotens
- Erweiterte Optionen

Erstellen einer Snowflake-Verbindung

> **Note**
>
> Vereinheitlichte Verbindungen (Verbindung v2) standardisieren alle zu verwendenden VerbindungenUSERNAME, PASSWORD Schlüssel für grundlegende Authentifizierungsdaten. Sie können immer noch eine v1-Verbindung über eine API mit Geheimnissen herstellen, die, enthaltensfUser. sfPassword

Beim Hinzufügen einer Datenquelle - Snowflake-Knoten in AWS Glue Studio, können Sie eine bestehende AWS Glue Snowflake-Verbindung auswählen oder eine neue Verbindung erstellen. Sie müssen einen SNOWFLAKE-Verbindungstyp auswählen und keinen JDBC-Verbindungstyp, der für die

Verbindung mit Snowflake konfiguriert ist. Gehen Sie wie folgt vor, um eine AWS Glue Snowflake-Verbindung herzustellen:

So erstellen Sie eine Snowflake-Verbindung

1. Generieren Sie in Snowflake einen Benutzer und ein Passwort. *snowflakeUser* *snowflakePassword*
2. Ermitteln Sie, mit welchem Snowflake-Warehouse dieser Benutzer interagieren wird,. *snowflakeWarehouse* Stellen Sie es entweder *snowflakeUser* in Snowflake als `DEFAULT_WAREHOUSE` für ein oder merken Sie es sich für den nächsten Schritt.
3. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit Ihren Snowflake-Anmeldeinformationen. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für *snowflakeUser* mit dem Schlüssel. `sfUser`
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für *snowflakePassword* mit dem Schlüssel. `sfPassword`
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für *snowflakeWarehouse* mit dem Schlüssel. `sfWarehouse` Dies ist nicht erforderlich, wenn in Snowflake ein Standardwert festgelegt ist.
4. Erstellen Sie im AWS Glue Datenkatalog eine Verbindung, indem Sie den Schritten unter Verbindung hinzufügen folgen. AWS Glue Behalten Sie nach dem Erstellen der Verbindung den Verbindungsnamen für den nächsten Schritt bei. *connectionName*
   - Wählen Sie bei der Auswahl eines Verbindungstyps Snowflake aus.
   - Geben Sie bei der Auswahl der Snowflake-URL den Hostnamen Ihrer Snowflake-Instance an. Die URL verwendet einen Hostnamen im Format *account_identifier*`.snowflakecomputing.com`.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*secretName*.

Erstellen eines Snowflake-Quellknotens

Berechtigungen erforderlich

AWS Glue Studio Jobs, die Snowflake-Datenquellen verwenden, erfordern zusätzliche Berechtigungen. Weitere Informationen zum Hinzufügen von Berechtigungen zu ETL-Aufträgen finden Sie unter Überprüfen der für ETL-Aufträge erforderlichen IAM-Berechtigungen.

`SNOWFLAKE` AWS Glue Verbindungen verwenden ein AWS Secrets Manager Geheimnis, um Anmeldeinformationen bereitzustellen. Ihre Job- und Datenvorschau-Rollen in AWS Glue Studio müssen berechtigt sein, dieses Geheimnis zu lesen.

Hinzufügen einer Snowflake-Datenquelle

Voraussetzungen:

- Ein AWS Secrets Manager Geheimnis für Ihre Snowflake-Anmeldeinformationen
- Eine Datenkatalogverbindung vom Typ Snowflake AWS Glue

So fügen Sie einen Knoten Datenquelle – Snowflake hinzu:

1. Wählen Sie die Verbindung für Ihre Snowflake-Datenquelle. Dabei wird davon ausgegangen, dass die Verbindung bereits besteht und Sie aus bestehenden Verbindungen auswählen können. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Snowflake-Verbindung erstellen. Weitere Informationen finden Sie unter Übersicht über die Verwendung von Konnektoren und Verbindungen.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken. Informationen zur Verbindung sind sichtbar, einschließlich URL, Sicherheitsgruppen, Subnetz, Verfügbarkeitszone, Beschreibung sowie erstellte (UTC) und letzte aktualisierte (UTC) Zeitstempel.
2. Wählen Sie eine Snowflake-Quellenoption:

   - Eine einzelne Tabelle auswählen – dies ist die Tabelle, die die Daten enthält, auf die Sie aus einer einzelnen Snowflake-Tabelle zugreifen möchten.
   - Benutzerdefinierte Abfrage eingeben – ermöglicht Ihnen den Zugriff auf einen Datensatz aus mehreren Snowflake-Tabellen basierend auf Ihrer benutzerdefinierten Abfrage.
3. Wenn Sie eine einzelne Tabelle ausgewählt haben, geben Sie den Namen eines Snowflake-Schemas ein.

Oder wählen Sie Benutzerdefinierte Abfrage eingeben aus. Wählen Sie diese Option, um aus mehreren Snowflake-Tabellen auf einen benutzerdefinierten Datensatz zuzugreifen. Wenn Sie diese Option auswählen, geben Sie die Snowflake-Abfrage ein.

4. Unter Leistungs- und Sicherheitsoptionen (optional)

   - Abfrage-Pushdown aktivieren – wählen Sie aus, wenn Sie die Arbeit an die Snowflake Instance auslagern möchten.

5. Geben Sie in den benutzerdefinierten Snowflake-Eigenschaften (optional) nach Bedarf Parameter und Werte ein.

Erstellen eines Snowflake-Zielknotens

Berechtigungen erforderlich

AWS Glue Studio Jobs, die Snowflake-Datenquellen verwenden, erfordern zusätzliche Berechtigungen. Weitere Informationen zum Hinzufügen von Berechtigungen zu ETL-Aufträgen finden Sie unter Überprüfen der für ETL-Aufträge erforderlichen IAM-Berechtigungen.

`SNOWFLAKE` AWS Glue Verbindungen verwenden ein AWS Secrets Manager Geheimnis, um Anmeldeinformationen bereitzustellen. Ihre Job- und Datenvorschau-Rollen in AWS Glue Studio müssen berechtigt sein, dieses Geheimnis zu lesen.

Hinzufügen eines Snowflake-Datenziels

So erstellen Sie einen Snowflake-Zielknoten:

1. Wählen Sie eine vorhandene Snowflake-Tabelle als Ziel oder geben Sie einen neuen Tabellennamen ein.
2. Wenn Sie den Zielknoten Datenziel – Snowflake verwenden, können Sie aus den folgenden Optionen wählen:

   - APPEND – Wenn eine Tabelle bereits vorhanden ist, übertragen Sie alle neuen Daten als Einfügung in diese Tabelle. Wenn die Tabelle nicht vorhanden ist, erstellen Sie sie und fügen Sie dann alle neuen Daten ein.
   - ZUSAMMENFÜHREN — AWS Glue aktualisiert Ihre Zieltabelle auf der Grundlage der von Ihnen angegebenen Bedingungen oder fügt Daten an sie an.

     Wählen Sie die Optionen aus:

- Schlüssel und einfache Aktionen auswählen – wählen Sie die Spalten, die als übereinstimmende Schlüssel zwischen den Quelldaten und Ihrem Zieldatensatz verwendet werden sollen.

  Geben Sie bei Übereinstimmung die folgenden Optionen an:

  - Aktualisieren Sie den Datensatz in Ihrem Zieldatensatz mit Daten aus der Quelle.
  - Löschen Sie den Datensatz in Ihrem Zieldatensatz.

  Geben Sie die folgenden Optionen an, wenn keine Übereinstimmung vorliegt:

  - Fügen Sie Quelldaten als neue Zeile in Ihren Zieldatensatz ein.
  - Nichts unternehmen.

- Eine benutzerdefinierte MERGE-Anweisung eingeben – Sie können dann Zusammenführungs-Anweisung validieren auswählen, um zu überprüfen, ob die Anweisung gültig oder ungültig ist.

- TRUNCATE – Wenn bereits eine Tabelle vorhanden ist, kürzen Sie die Tabellendaten, indem Sie zunächst den Inhalt der Zieltabelle löschen. Wenn das Kürzen erfolgreich ist, fügen Sie alle Daten ein. Wenn die Tabelle nicht vorhanden ist, erstellen Sie die Tabelle und fügen Sie alle Daten ein. Wenn das Kürzen nicht erfolgreich ist, schlägt der Vorgang fehl.
- DROP – Wenn eine Tabelle bereits vorhanden ist, löschen Sie die Tabellenmetadaten und -daten. Wenn der Löschvorgang erfolgreich ist, fügen Sie alle Daten ein. Wenn die Tabelle nicht vorhanden ist, erstellen Sie die Tabelle und fügen Sie alle Daten ein. Wenn das Löschen nicht erfolgreich ist, schlägt der Vorgang fehl.

Erweiterte Optionen

Weitere Informationen finden Sie im Entwicklerhandbuch unter Snowflake-Verbindungen. AWS Glue

Verbindung zu Stripe herstellen in AWS Glue Studio

Stripe ist eine Plattform für Online-Zahlungsabwicklung und Kreditkartenabwicklung für Unternehmen. Die Stripe-Plattform ermöglicht es Unternehmen, Online-Zahlungen zu akzeptieren, ein Abonnement (wiederkehrende Abrechnung) für ihren E-Commerce zu erstellen und ein Backkonto für den Empfang von Auszahlungen einzurichten. Stripe unterstützt auch Zahlungen an mehrere Parteien, sodass Unternehmen ihren eigenen Marktplatz einrichten und Zahlungen einziehen und dann über ein „Connected“ -Konto an Verkäufer oder Dienstleister auszahlen können.

Themen

- AWS Glue Unterstützung für Stripe
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Stripe konfigurieren
- Konfiguration von Stripe-Verbindungen
- Aus Stripe-Entitäten lesen
- Stripe-Verbindungsoptionen
- Einschränkungen
- Neues Stripe-Konto erstellen und die Client-App konfigurieren

AWS Glue Unterstützung für Stripe

AWS Glue unterstützt Stripe wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Stripe abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Slack-API-Versionen

v1.

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Sie können auch die folgenden verwalteten IAM-Richtlinien verwenden, um den Zugriff zu ermöglichen:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM und Amazon EC2. Amazon CloudWatch Logs Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Stripe konfigurieren

Bevor Sie Daten von Stripe AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

- Sie müssen über ein Stripe-Konto mit E-Mail und Passwort verfügen. Weitere Informationen finden Sie unter Neues Stripe-Konto erstellen und die Client-App konfigurieren.
- Ihr Stripe-Konto ist für den API-Zugriff aktiviert. Jegliche Nutzung der Stripe-API ist ohne zusätzliche Kosten möglich.

Wenn Sie diese Anforderungen erfüllen, können Sie sich mit Ihrem Stripe-Konto verbinden AWS Glue .

Konfiguration von Stripe-Verbindungen

Stripe unterstützt die benutzerdefinierte Authentifizierung. Weitere Informationen zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie in der STRIPE REST API-Dokumentation.

So konfigurieren Sie eine Stripe-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit den folgenden Details. Es ist erforderlich, für jede Verbindung in ein Geheimnis zu erstellen AWS Glue.
   a. Für vom Kunden verwaltete verbundene Apps sollte Secret die verbundene App Consumer Secret mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Stripe aus.
   b. Wählen Sie die IAM-Rolle aus, die folgende Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

c. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

d. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Stripe-Entitäten lesen

Voraussetzungen

- Ein Stripe-Objekt, aus dem Sie lesen möchten.

Unterstützte Entitäten

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Ausgewogenheit	Nein	Nein	Nein	Ja	Nein
Transaktionen ausgleichen	Ja	Ja	Nein	Ja	Ja
Gebühren	Ja	Ja	Nein	Ja	Ja
Streitfälle	Ja	Ja	Nein	Ja	Ja
Links zu Dateien	Ja	Ja	Nein	Ja	Ja
PaymentIntents	Ja	Ja	Nein	Ja	Ja
SetupIntents	Ja	Ja	Nein	Ja	Ja
Auszahlungen	Ja	Ja	Nein	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Erstattungen	Ja	Ja	Nein	Ja	Ja
Produkte	Ja	Ja	Nein	Ja	Ja
Preise	Ja	Ja	Nein	Ja	Ja
Gutscheine	Ja	Ja	Nein	Ja	Ja
Promotion-Codes	Ja	Ja	Nein	Ja	Ja
Steuercodes	Nein	Ja	Nein	Ja	Nein
Steuersätze	Ja	Ja	Nein	Ja	Ja
Versandtarife	Ja	Ja	Nein	Ja	Ja
Sitzungen	Ja	Ja	Nein	Ja	Ja
Gutschriften	Ja	Ja	Nein	Ja	Ja
Customer	Ja	Ja	Nein	Ja	Ja
Rechnungen	Ja	Ja	Nein	Ja	Ja
Rechnungsposten	Ja	Ja	Nein	Ja	Nein
Pläne	Ja	Ja	Nein	Ja	Ja
Zitate	Ja	Ja	Nein	Ja	Nein
Subscriptions (Abonnements)	Ja	Ja	Nein	Ja	

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order By	Unterstützt Select *	Unterstützt Partitionierung
Artikel im Abonnement	Nein	Ja	Nein	Ja	Nein
Abonnement-Zeitpläne	Ja	Ja	Nein	Ja	Ja
Konten	Nein	Ja	Nein	Ja	Ja
Gebühren für die Bewerbung	Ja	Ja	Nein	Ja	Ja
Länderspezifikationen	Nein	Ja	Nein	Ja	Nein
Weiterleitungen	Ja	Ja	Nein	Ja	Ja
Frühe Betrugswarnungen	Ja	Ja	Nein	Ja	Ja
Berichtstypen	Nein	Nein	Nein	Ja	Nein

Beispiel

```
stripe_read = glueContext.create_dynamic_frame.from_options(
    connection_type="stripe",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1"
    }
)
```

Angaben zur Stripe-Entität und zum Feld

Entität	Feld	Datentyp	Unterstützte Operatoren
Saldo			
	available	Auflisten	
	connect_reserved	Auflisten	
	ausstehend	Auflisten	
	Live-Modus	Boolesch	
	object	String	
	sofort verfügbar	Auflisten	
	ausstellend	Struct	
Transaktionen ausgleichen			
	id	String	
	object	String	
	Betrag	Ganzzahl	
	verfügbar_am	DateTime	=, >=, <=,<,>
	created	DateTime	=, >=, <=,<,>
	currency	String	
	description	String	
	wechselkurs	BigDecimal	
	Gebühr	Ganzzahl	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Gebührendetails	Auflisten	
	net	Ganzzahl	
	Berichtskategorie	String	
	Quelle	String	=
	Status	String	
	Typ	String	=
	grenzübergreifende Klassifizierung	String	
Gebühren			
	id	String	
	object	String	
	Betrag	Ganzzahl	=, <, >
	erfasster Betrag	Ganzzahl	
	betrag_rückerstattet	Ganzzahl	
	Anwendung	String	
	Anmeldegebühr	String	
	Höhe der Antragsgebühr	Ganzzahl	
	Saldo_Transaktion	String	
	Abrechnungsdetails	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Deskriptor für berechnete_Angaben	String	
	gefangen	Boolesch	
	created	DateTime	=, >=, <=,<,>
	currency	String	
	customer	String	=
	description	String	
	Ziel	String	
	Streit	String	
	umstritten	Boolesch	=
	failure_balance_transaction	String	
	Fehlercode	String	
	Störungsmeldung	String	
	Einzelheiten zum Betrug	Struct	
	Rechnung	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	im Namen von	String	
	order	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Ergebnis	Struct	
	bezahlt	Boolesch	
	Zahlungsabsicht	String	=
	Zahlungsmethode	String	
	Einzelheiten zur Zahlungsmethode	Struct	
	Empfangs-E-Mail	String	
	Empfangsnummer	String	
	Empfangs-URL	String	
	erstattet	Boolesch	=
	Rückerstattungen	Struct	
	Rezension	String	
	Versand	Struct	
	Quelle	Struct	
	source_transfer	String	
	Anweisungsdeskriptor	String	
	Suffix für Anweisungsbeschreibungen	String	
	Status	String	
	Übertragung	String	
	Daten übertragen	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	transfer_group	String	=
Streitfälle			
	id	String	
	object	String	
	Betrag	Ganzzahl	=, <, >
	balance_transaction	String	
	balance_transaktionen	Auflisten	
	aufladen	String	=
	created	DateTime	=, >=, <=,<,>
	currency	String	
	Beweismittel	Struct	
	Beweisdetails	Struct	
	ist_gebühren_erstattungsfähig	Boolesch	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Zahlungsabsicht	String	=
	Grund	String	=
	Status	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Einzelheiten zur Zahlungsmethode	Struct	
Links zu Dateien			
	id	String	
	object	String	
	created	DateTime	=, >=, <=,<,>
	abgelaufen	Boolesch	=
	expires_at	DateTime	
	file	String	=
	Live-Modus	Boolesch	
	Metadaten	Struct	
	URL	String	
PaymentIntents			
	id	String	
	object	String	
	Betrag	Ganzzahl	
	erfassbarer Betrag	Ganzzahl	
	Angaben zur Menge	Struct	
	erhaltener Betrag	Ganzzahl	
	Anwendung	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Höhe der Antragsgebühr	Ganzzahl	
	automatische_Zahlungsmethoden	Struct	
	storniert_am	DateTime	
	Grund der Stornierung	String	
	Erfassungsmethode	String	
	client_secret	String	
	Bestätigungsmethode	String	
	created	DateTime	=, >=, <=,<,>
	currency	String	
	customer	String	=
	description	String	
	Rechnung	String	
	letzter Zahlungsfehler	Struct	
	neueste_Gebühr	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	nächste_Aktion	Struct	
	im Namen von	String	
	Zahlungsmethode	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Zahlungsmethodenoptionen	Struct	
	Arten von Zahlungsmethoden	Auflisten	
	Einzelheiten zur Konfiguration der Zahlungsmethode	Struct	
	Verarbeitung	Struct	
	Empfangs-E-Mail	String	
	Rezension	String	
	setup_future_usage	String	
	Versand	Struct	
	Quelle	String	
	statement_descriptor	String	
	Suffix für Anweisungsbeschreibungen	String	
	Status	String	
	Daten übertragen	Struct	
	transfer_group	String	
SetupIntents			
	id	String	
	object	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Anwendung	String	
	Grund der Stornierung	String	
	client_secret	String	
	created	DateTime	=, >=, <=,<,>
	customer	String	=
	description	String	
	Flow_Directions	Auflisten	
	letzter Einrichtungsfehler	Struct	
	neuestes_Versuch	String	
	Live-Modus	Boolesch	
	Mandat	String	
	Metadaten	Struct	
	nächste_Aktion	Struct	
	im Namen von	String	
	Zahlungsmethode	String	
	Zahlungsmethodenoptionen	Struct	
	Arten von Zahlungsmethoden	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Mandat zur einmaligen Verwendung	String	
	Status	String	
	Nutzung	String	
	automatische_Zahlungsmethoden	Struct	
Auszahlungen			
	id	String	
	object	String	
	Betrag	Ganzzahl	=, <, >
	Anreisedatum	DateTime	=, >=, <=,<,>
	Automatisch	Boolesch	
	Saldo_Transaktion	String	
	created	DateTime	=, >=, <=,<,>
	currency	String	
	description	String	=
	Ziel	String	
	fehlgeschlagene_balance_transaktion	String	
	Fehlercode	String	
	Störungsmeldung	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Methode	String	
	ursprüngliche_Auszahlung	String	
	umgekehrt_von	String	
	Status der Versöhnung	String	
	source_type	String	
	Aussagedeskriptor	String	
	Status	String	
	Typ	String	
	Bewerbungsgebühr	String	
	Höhe der Antragsgebühr	Ganzzahl	
Erstattungen			
	id	String	
	object	String	
	Betrag	Ganzzahl	
	Saldo_Transaktion	String	
	aufladen	String	=

Entität	Feld	Datentyp	Unterstützte Operatoren
	created	DateTime	=, >=, <=,<,>
	currency	String	
	Metadaten	Struct	
	Angaben zum Reiseziel	Struct	
	Zahlungsabsicht	String	=
	Grund	String	
	Quittungsnummer	String	
	Quelle_Transfer_Umkehrung	String	
	Status	String	
	Umkehrung der Übertragung	String	
Produkte			
	id	String	
	object	String	
	aktiv	Boolesch	=
	Attribute	Auflisten	
	created	DateTime	=, >=, <=,<,>
	Standardpreis	String	
	description	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Bilder	Auflisten	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Name	String	
	Abmessungen des Pakets	Struct	
	versandfähig	Boolesch	
	aussagekräfte_deskriptor	String	
	Steuernummer	String	
	Typ	String	=
	Bezeichnung der Einheit	String	
	Aktualisiert	DateTime	
	URL	String	
	features	Auflisten	
Preise			
	id	String	
	object	String	
	aktiv	Boolesch	=
	Abrechnungssystem	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	created	DateTime	=, >=, <=,<,>
	currency	String	=
	benutzerdefinierte r_Einheitsbetrag	Struct	
	Live-Modus	Boolesch	
	Suchschlüssel	String	
	Metadaten	Struct	
	Spitzname	String	
	Produkt	String	=
	wiederkehrend	Struct	
	steuerliches Verhalten	String	
	Tiers-Modus	String	
	Menge transformieren	Struct	
	Typ	String	=
	Stückzahl	Ganzzahl	
	Einheit_Betrag_Dez imal	String	
Gutscheine			
	Id	String	
	object	String	
	Betrag_Rabatt	Ganzzahl	

Entität	Feld	Datentyp	Unterstützte Operatoren
	created	DateTime	=, >=, <=,<,>
	currency	String	=
	duration	String	=
	Dauer_in Monaten	Ganzzahl	=,<,>
	Live-Modus	Boolesch	
	max_redemption	Ganzzahl	=, <, >
	Metadaten	Struct	
	Name	String	
	prozentualer Preisnachlass	Double	=
	einlösen von	DateTime	=, >=, <=, <, >
	times_eingelöst	Ganzzahl	
	valid	Boolesch	
Aktionscodes			
	Id	String	
	object	String	
	aktiv	Boolesch	=
	Code	String	=
	Coupon	Struct	
	created	DateTime	=,>=,<=,<,>

Entität	Feld	Datentyp	Unterstützte Operatoren
	customer	String	
	läuft ab_at	DateTime	
	Live-Modus	Boolesch	
	max_redemption	Ganzzahl	
	Metadaten	Struct	
	Einschränkungen	Struct	
	mal eingelöst	Ganzzahl	
Steuerkennzeichen			
	Id	String	
	object	String	
	description	String	
	Name	String	
Steuersätze			
	Id	String	
	object	String	
	aktiv	Boolesch	=
	country	String	
	created	DateTime	=, >=, <=, <, >
	description	String	
	Anzeige_name	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	einschließlich	Boolesch	=
	Zuständigkeit	String	
	jurisdiction_level	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	percentage	Double	
	effektiver_Prozentsatz	Double	
	state	String	
	Steuerart	String	
Versandtarife			
	Id	String	
	object	String	
	aktiv	Boolesch	=
	created	DateTime	=, >=, <=, <, >
	Delivery_Estimate	Struct	
	Anzeige_name	String	
	festgelegter_Betrag	Struct	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	steuerliches Verhalten	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Steuergesetzbuch	String	
	Typ	String	
Sitzungen			
	id	String	
	object	String	
	nach_Ablauf	Struct	
	Promotionscodes zulassen	Boolesch	
	Betrag_Zwischensumme	Ganzzahl	
	Betrag_Summe	Ganzzahl	
	automatische_Steuer	Struct	
	Erfassung der Rechnungsadresse	String	
	URL stornieren	String	
	Client-Referenz-ID	String	
	Zustimmung	Struct	
	Einholung von Einwilligungserklärungen	Struct	
	created	DateTime	=, >=, <=, <, >
	currency	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	benutzerdefinierter_Text	Struct	
	customer	String	
	Kundenerstellung	String	
	Kundendetails	Struct	
	Kunden-E-Mail	String	
	läuft ab_am	DateTime	
	Rechnung	String	
	Rechnungserstellung	Struct	
	Live-Modus	Boolesch	
	locale	String	
	Metadaten	Struct	
	mode	String	
	Zahlungsabsicht	String	=
	Zahlungslink	String	
	Zahlungsmethode_Inkasso	String	
	Zahlungsmethodenoptionen	Struct	
	Arten von Zahlungsmethoden	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Zahlungsstatus	String	
	Telefonnummer_Sammlung	Struct	
	wiederhergestellt_von	String	
	setup_intent	String	
	Versandadresse/Abholung	Struct	
	versandkosten	Struct	
	Versanddetails	Struct	
	Versandoptionen	Auflisten	
	Status	String	
	einreichen_	String	
	Abonnement	String	
	Erfolgs-URL	String	
	Sammlung von Steuer-Identifikationsnummern	Struct	
	insgesamte_Angaben	Struct	
	URL	String	
	ui-Modus	String	
Gutschriften			

Entität	Feld	Datentyp	Unterstützte Operatoren
	id	String	
	object	String	
	Betrag	Ganzzahl	
	created	DateTime	=, >=, <=, <, >
	currency	String	
	customer	String	=
	customer_balance_transaction	String	
	Rabattbetrag	Ganzzahl	
	Rabattbeträge	Auflisten	
	Rechnung	String	=
	lines	Struct	
	Live-Modus	Boolesch	
	Memo	String	
	Metadaten	Struct	
	Zahl	String	
	un_von_band_amount	Ganzzahl	
	pdf	String	
	Grund	String	
	Erstattung	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Status	String	
	Zwischensumme	Ganzzahl	
	Zwischensumme ohne Steuern	Ganzzahl	
	Steuerbeträge	Auflisten	
	total	Ganzzahl	
	insgesamt ohne Steuern	Ganzzahl	
	Typ	String	
	nichtig_am	DateTime	
	Menge_Versand	Ganzzahl	
	effektiv_am	DateTime	
	versandkosten	Struct	
Customer			
	id	String	
	object	String	
	address	Struct	
	balance	Ganzzahl	
	created	DateTime	
	currency	String	=, >=, <=, <, >
	Standardquelle	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Delinquent	Boolesch	=
	description	String	
	discount	Struct	
	email	String	=
	invoice_prefix	String	
	Rechnungseinstellungen	Struct	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Name	String	
	nächste_Rechnungssequenz	Ganzzahl	
	phone	String	
	bevorzugte_Gebietsschemas	Auflisten	
	Versand	Struct	
	tax_exempt	String	
	test_clock	String	
Rechnungen			
	id	String	
	object	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Land des Benutzerkontos	String	
	Konto_Bezeichnung	String	
	Kontonumsatzsteuer-IDs	Auflisten	
	fälliger Betrag	Ganzzahl	
	bezahlter Betrag	Ganzzahl	
	verbleibender Betrag	Ganzzahl	
	Anwendung	String	
	Höhe der Antragsgebühr	Ganzzahl	
	Anzahl der Versuche	Ganzzahl	
	versucht	Boolesch	=
	auto_advance	Boolesch	=
	automatische_Steuer	Struct	
	Abrechnungsgrund	String	
	aufladen	String	
	Erhebungsmethode	String	=
	created	DateTime	=, >=, <=, <, >
	currency	String	
	custom_fields	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	customer	String	=
	customer_address	Struct	
	E-Mail-Adresse des Kunden	String	
	Kundenname	String	
	Telefonnummer des Kunden	String	
	kunden_versand	Struct	
	kunden_steuerbefreit	String	
	Steuer-IDs von Kunden	Auflisten	
	standardmäßige Zahlungsmethode	String	
	Standardquelle	String	
	default_tax_rates	Auflisten	
	description	String	
	discount	Struct	
	Rabatte	Auflisten	
	due_date	DateTime	=, >=, <=, <, >
	ending_balance	Ganzzahl	
	Fußzeile	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	von_invoice	Struct	
	URL für gehostete Rechnungen	String	
	Rechnung_PDF	String	
	letzter Finalisierungsfehler	Struct	
	neueste_Revision	String	
	lines	Struct	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	nächster_Zahlungsversuch	DateTime	
	Zahl	String	
	im Namen von	String	
	bezahlt	Boolesch	=
	ausnahmslos bezahlt	Boolesch	
	Zahlungsabsicht	String	
	Zahlungseinstellungen	Struct	
	Ende des Zeitraums	DateTime	=, >=, <=, <, >
	Periodenbeginn	DateTime	=, >=, <=, <, >

Entität	Feld	Datentyp	Unterstützte Operatoren
	Betrag der Gutschriften nach erfolgter Zahlung	Ganzzahl	
	Betrag der vorausbezahlten Gutschriften	Ganzzahl	
	Zitat	String	
	Quittungsnummer	String	
	Rendern	Struct	
	Renderoptionen	Struct	
	Startbilanz	Ganzzahl	
	Ausweisungsdeskriptor	String	
	Status	String	=
	status_transitions	Struct	
	Abonnement	String	
	Einzelheiten zum Abonnement	Struct	
	Zwischensumme	Ganzzahl	=, <, >
	Zwischensumme ohne Steuern	Ganzzahl	
	Steuer	Ganzzahl	
	test_clock	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	total	Ganzzahl	=, <, >
	Gesamtzahl der Rabattbeträge	Auflisten	
	insgesamt ohne Steuern	Ganzzahl	
	Gesamtzahl der Steuerbeträge	Auflisten	
	Daten übertragen	Struct	
	Webhooks_geliefert_unter	DateTime	
	finalisiert_at automatisch_	DateTime	
	effektiv_am	DateTime	
	Aussteller	Struct	
Rechnungsposten			
	id	String	
	object	String	
	Betrag	Ganzzahl	=, <, >
	currency	String	
	customer	String	=
	date	DateTime	
	description	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	rabattierbar	Boolesch	
	Rabatte	Auflisten	
	Rechnung	String	=
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Zeitraum	Struct	
	Planen	Struct	
	price	Struct	
	Aufteilung	Boolesch	=
	quantity	Ganzzahl	
	Abonnement	String	
	Abonnement-Artikel	String	
	steuersätze	Auflisten	
	test_clock	String	
	Anzahl pro Einheit	Ganzzahl	
	Einheit_Betrag_Dezimal	String	
Pläne			
	id	String	
	object	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	aktiv	Boolesch	=
	aggregate_nutzung	String	
	Betrag	Ganzzahl	
	Menge_Dezimal	String	
	Abrechnungsschema	String	
	created	DateTime	=, >=, <=, <, >
	currency	String	=
	Intervall	String	=
	Anzahl der Intervalle	Ganzzahl	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Spitzname	String	
	Produkt	String	=
	Tiers_Mode	String	
	Nutzung transform ieren	Struct	
	Testzeitraum/Tage	Ganzzahl	=, <, >
	Nutzungstyp	String	
	Gebührenerfassung	String	
Zitate			

Entität	Feld	Datentyp	Unterstützte Operatoren
	id	String	
	object	String	
	Betrag_Zwischensumme	Ganzzahl	
	Betrag_Summe	Ganzzahl	
	Anwendung	String	
	Höhe der Antragsgebühr	Ganzzahl	
	Antragsgebühr_Prozent	Double	
	automatische_Steuer	Struct	
	Erhebungsmethode	String	
	berechnet	Struct	
	created	DateTime	
	currency	String	
	customer	String	=
	default_tax_rates	Auflisten	
	description	String	
	Rabatte	Auflisten	
	läuft ab_at	DateTime	
	Fußzeile	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	von_quote	Struct	
	header	String	
	Rechnung	String	
	Rechnungseinstellungen	Struct	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Zahl	String	
	im Namen von	String	
	Status	String	=
	status_transitions	Struct	
	Abonnement	String	
	Abonnementdaten	Struct	
	Abonnementplan	String	
	test_clock	String	
	insgesamte_Einzelheiten	Struct	
	Daten übertragen	Struct	
Subscriptions (Abonnements)			
	id	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	object	String	
	Anwendung	String	
	Antragsgebühr_Prozent	Double	
	automatische_Steuer	Struct	
	Anchor für den Abrechnungszyklus	DateTime	
	Abrechnungsschwellenwerte	Struct	
	cancel_at	DateTime	
	am Ende des Zeitraums stornieren	Boolesch	
	storniert_am	DateTime	
	Erfassungsmethode	String	=
	created	DateTime	=, >=, <=,<,>
	currency	String	
	Ende des aktuellen Zeitraums	DateTime	=, >=, <=
	Beginn des aktuellen Zeitraumes	DateTime	=, >=, <=
	customer	String	=
	Tage_bis zur Fälligkeit	Ganzzahl	

Entität	Feld	Datentyp	Unterstützte Operatoren
	standardmäßige Zahlungsmethode	String	
	Standardquelle	String	
	default_tax_rates	Auflisten	
	description	String	
	discount	Struct	
	ended_at	DateTime	
	Elemente	Struct	
	neueste_Rechnung	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Rechnung mit der nächsten ausstehen den Rechnung	DateTime	
	Erfassung pausieren	Struct	
	Zahlungseinstellun gen	Struct	
	Intervall für ausstehen de Rechnungen	Struct	
	ausstehend_setup_i ntent	String	
	ausstehend_aktuali sierung	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Planen	Struct	
	quantity	Ganzzahl	
	schedule	String	
	Startdatum	DateTime	
	Status	String	=
	test_clock	String	
	Daten übertragen	Struct	
	Ende der Testphase	DateTime	
	Teststart	DateTime	
Artikel im Abonnement			
	Id	String	
	object	String	
	billing_thresholds	Struct	
	created	DateTime	=, >=, <=, <, >
	Metadaten	Struct	
	Planen	Struct	
	price	Struct	
	Abonnement	String	
	Steuersätze	Auflisten	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Rabatte	Auflisten	
Abonnement-Zeitpläne			
	object	String	
	Anwendung	String	
	storniert_at	DateTime	
	abgeschlossen_am	DateTime	
	created	DateTime	
	aktuelle_Phase	Struct	
	customer	String	=
	Standardeinstellungen	Struct	
	Endverhalten	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	phases	Auflisten	
	veröffentlicht_am	DateTime	
	veröffentlichtes_Abonnement	String	
	Verlängerungsintervall	String	
	Status	String	
	Abonnement	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	test_clock	String	
Konten			
	details_eingereicht	Boolesch	
	tos_acceptance	Struct	
	Typ	String	
	Metadaten	Struct	
	id	String	
	object	String	
	Standardwährung	String	
	Funktionen	Struct	
	charges_aktiviert	Boolesch	
	Einstellungen	Struct	
	Anforderungen	Struct	
	Auszahlungen_aktiviert	Boolesch	
	Zukünftige_Anforderungen	Struct	
	externe_Konten	Struct	
	Controller	Struct	
	country	String	
	email	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	created	DateTime	=, >=, <=, <, >
	Unternehmensprofil	Struct	
	Geschäftstyp	String	
	company	Struct	
Gebühren für die Bewerbung			
	id	String	
	object	String	
	Konto	String	
	Betrag	Ganzzahl	=, <, >
	Betrag zurückerstattet	Ganzzahl	=, <, >
	Anwendung	String	
	saldo_transaktion	String	
	aufladen	String	=
	created	DateTime	
	currency	String	
	Live-Modus	Boolesch	
	originierende_Transaktion	String	
	erstattet	Boolesch	=
	Rückerstattungen	Struct	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Gebührenquelle	Struct	
Technische Daten des Landes			
	id	String	
	object	String	
	Standardwährung	String	
	Unterstützte_Bankkontowährungen	Struct	
	Unterstützte_Zahlungswährungen	Auflisten	
	unterstützte_Zahlungsmethoden	Auflisten	
	unterstützte_Transferländer	Auflisten	
	Bestätigungsfelder	Struct	
Weiterleitungen			
	id	String	
	object	String	
	Betrag	Ganzzahl	=, <, >
	Betrag_rückgängig gemacht	Ganzzahl	
	salden_transaktion	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	created	DateTime	=, >=, <=, <, >
	currency	String	=
	description	String	
	Ziel	String	=
	Ziel_Zahlung	String	
	Live-Modus	Boolesch	
	Metadaten	Struct	
	Umkehrungen	Struct	
	rückgängig gemacht	Boolesch	
	Quelltransaktion	String	
	source_type	String	
	Transfergruppe	String	=
Frühe Betrugswarnungen			
	id	String	
	object	String	
	umsetzbar	Boolesch	
	aufladen	String	=
	created	DateTime	=, >=, <=, <, >
	Art des Betrugs	String	

Entität	Feld	Datentyp	Unterstützte Operatoren
	Live-Modus	Boolesch	
	Zahlungsabsicht	String	=
Berichtstypen			
	id	String	
	object	String	
	data_available_end	DateTime	
	data_available_start	DateTime	
	Standardspalten	Auflisten	
	Live-Modus	Boolesch	
	Name	String	
	Aktualisiert	DateTime	
	version	Ganzzahl	

Abfragen partitionieren

Zusätzliche Spark-Optionen`PARTITION_FIELD`,`LOWER_BOUND`,`UPPER_BOUND`, `NUM_PARTITIONS` können bereitgestellt werden, wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- `PARTITION_FIELD`: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Als Datum akzeptieren wir das Spark-Datumsformat, das in Spark-SQL-Abfragen verwendet wird. Beispiel für einen gültigen Wert:`"2024-07-01T00:00:00.000Z"`.

- `UPPER_BOUND`: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`: Anzahl der Partitionen.

Einzelheiten zur Unterstützung von Feldern für die entitätsweise Partitionierung sind in der folgenden Tabelle aufgeführt.

Entity Name	Feld partitionieren	Datentyp
Transaktionen ausgleichen	created	DateTime
Gebühren	created	DateTime
Streitfälle	created	DateTime
Links zu Dateien	created	DateTime
PaymentIntents	created	DateTime
SetupIntents	created	DateTime
Auszahlungen	created	DateTime
Erstattungen	created	DateTime
Produkte	created	DateTime
Preise	created	DateTime
Gutscheine	created	DateTime
Promotion-Codes	created	DateTime
Steuersätze	created	DateTime
Versandtarife	created	DateTime
Sitzungen	created	DateTime
Gutschriften	created	DateTime
Customer	created	DateTime

Entity Name	Feld partitionieren	Datentyp
Rechnungen	created	DateTime
Pläne	created	DateTime
Subscriptions (Abonnements)	created	DateTime
Abonnement-Zeitpläne	created	DateTime
Konten	created	DateTime
Gebühren für die Bewerbung	created	DateTime
Weiterleitungen	created	DateTime
Frühe Betrugswarnungen	created	DateTime

Beispiel

```
stripe_read = glueContext.create_dynamic_frame.from_options(
    connection_type="stripe",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "created"
        "LOWER_BOUND": "2024-05-01T20:55:02.000Z"
        "UPPER_BOUND": "2024-07-11T20:55:02.000Z"
        "NUM_PARTITIONS": "10"
    }
)
```

Stripe-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Stripe aufgeführt:

- `ENTITY_NAME`(String) — (Erforderlich) Wird für Lesen/Schreiben verwendet. Der Name Ihres Objekts in Stripe.

- `API_VERSION`(String) — (Erforderlich) Wird zum Lesen/Schreiben verwendet. Stripe Rest API-Version, die Sie verwenden möchten. Beispiel: v1.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Einschränkungen

Die folgenden Einschränkungen gelten für den Stripe-Connector:

- Nur die feldbasierte Partitionierung wird vom Connector unterstützt.
- Die datensatzbasierte Partitionierung wird vom Connector nicht unterstützt, es besteht keine Möglichkeit, die Gesamtzahl der Datensätze abzurufen.
- Der Primärschlüssel-Datentyp ist „Zeichenfolge“, sodass die ID-basierte Partitionierung nicht nach Konnektoren unterstützt wird.

Neues Stripe-Konto erstellen und die Client-App konfigurieren

Ein Stripe-Konto erstellen

1. Wählen Sie auf dem Link https://dashboard.stripe.com/registrieren.
2. Geben Sie Ihre E-Mail-Adresse, Ihren vollständigen Namen und Ihr Passwort ein und wählen Sie Konto erstellen.

3. Nachdem Sie sich mit dem Konto angemeldet haben, verifizieren Sie das Konto, indem Sie Gmail öffnen wählen.
4. Bestätigen Sie das Konto, indem Sie auf den Bestätigungslink klicken, den Sie per E-Mail erhalten haben.
5. Nachdem Sie auf E-Mail-Adresse bestätigen geklickt haben, wird auf eine andere Seite weitergeleitet
6. Nachdem Sie auf Zahlungen aktivieren geklickt haben, um das Konto zu aktivieren, wird es zur Seite Zahlungen aktivieren (https://dashboard.stripe.com/welcome) weitergeleitet. Stellen Sie sicher, dass Sie alle Ihre gültigen Daten eingegeben haben, und klicken Sie anschließend auf die Schaltfläche Weiter.

Eine Slack-Entwickler-App erstellen

1. Loggen Sie sich bei Stripe ein.
2. Wählen Sie Entwickler aus, wie im Bild unten ganz oben angezeigt.
3. Wählen Sie unter Entwickler die Option API-Schlüssel aus.
4. Wählen Sie Testschlüssel anzeigen, um die API-Schlüssel abzurufen.

Verbindung zu Teradata Vantage herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für Teradata Vantage. AWS Glue Studio bietet eine visuelle Oberfläche, über die Sie eine Verbindung zu Teradata herstellen, Datenintegrationsjobs erstellen und diese auf dem ausführen können AWS Glue Studio serverlose Spark-Laufzeit.

AWS Glue Studio erstellt eine einheitliche Verbindung für Teradata Vantage. Weitere Informationen finden Sie unter Überlegungen.

Themen

- Erstellen einer Teradata-Vantage-Verbindung
- Erstellen eines Teradata-Quellknotens
- Erstellen eines Teradata-Zielknotens
- Erweiterte Optionen

Erstellen einer Teradata-Vantage-Verbindung

Um von aus eine Verbindung zu Teradata Vantage herzustellen AWS Glue, müssen Sie Ihre Teradata-Anmeldeinformationen erstellen und in einem AWS Secrets Manager Secret speichern und dieses Secret dann mit einer Teradata-Verbindung verknüpfen. AWS Glue

Voraussetzungen:

- Wenn Sie über Amazon VPC auf Ihre Teradata-Umgebung zugreifen, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue Job mit der Teradata-Umgebung kommunizieren kann. Wir raten davon ab, über das öffentliche Internet auf die Teradata-Umgebung zuzugreifen.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die bei der Ausführung des Jobs verwendet AWS Glue werden. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer Teradata-Instance und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem Teradata-Client-Port herstellen. Weitere Informationen zu Teradata-Ports finden Sie in der Teradata-Dokumentation.

  Abhängig von Ihrem Netzwerklayout kann eine sichere VPC-Konnektivität Änderungen an Amazon VPC und anderen Netzwerkdiensten erfordern. Weitere Informationen zur Konnektivität finden Sie unter AWS Konnektivitätsoptionen in der AWS Teradata-Dokumentation.

So konfigurieren Sie eine AWS Glue Teradata-Verbindung:

1. Identifizieren oder erstellen Sie in Ihrer Teradata-Konfiguration einen Benutzer und ein Passwort, mit dem eine Verbindung hergestellt AWS Glue wird, und. *`teradataUser`* *`teradataPassword`* Weitere Informationen finden Sie in der Teradata-Dokumentation unter Vantage Security Overview.
2. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer Teradata-Anmeldeinformationen ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *`teradataUsername`*

- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *`teradataPassword`*

3. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *`connectionName`*

   - Wählen Sie als Verbindungstyp Teradata aus.
   - Wenn Sie die IDBC-URL angeben, geben Sie die URL für Ihre Instance an. Sie können auch bestimmte durch Kommas getrennte Verbindungsparameter in Ihrer JDBC-URL fest codieren. Die URL muss dem folgenden Format entsprechen: `jdbc:teradata://`*`teradataHostname`*`/`*`ParameterName`*`=`*`ParameterValue`*`,`*`ParameterName`*

     Zu den unterstützten URL-Parametern gehören:

     - `DATABASE` – Name der Datenbank auf dem Host, auf die standardmäßig zugegriffen werden soll.
     - `DBS_PORT` – der Datenbankport, der verwendet wird, wenn nicht der standardmäßige Port genutzt wird.
   - Wählen Sie bei der Auswahl eines Anmeldeinformationstyps die Option AWS Secrets Managerund setzen Sie dann AWS Geheim auf*`secretName`*.

4. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für Teradata-Instances, die AWS in einer Amazon VPC gehostet werden
     - Sie müssen Amazon VPC-Verbindungsinformationen für die Verbindung bereitstellen, die Ihre AWS Glue Teradata-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Erstellen eines Teradata-Quellknotens

Voraussetzungen

- Eine AWS Glue Teradata Vantage-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,. the section called “Erstellen einer Teradata-Vantage-Verbindung”
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.

- Eine Teradata-Tabelle, aus der Sie lesen, oder abfragen möchten. *tableName targetQuery*

Hinzufügen einer Teradata-Datenquelle

Einen Knoten Datenquelle – Teradata fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für die Teradata-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Neue Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Teradata-Vantage-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie eine Option für Teradata-Quelle aus:
   - Eine einzelne Tabelle auswählen – rufen Sie alle Daten aus einer einzigen Tabelle ab.
   - Benutzerdefinierte Abfrage eingeben – rufen Sie einen Datensatz aus mehreren Tabellen ab, basierend auf Ihrer benutzerdefinierten Abfrage.
3. Wenn Sie eine einzelne Tabelle ausgewählt haben, geben Sie ein. *tableName*

   Wenn Sie Benutzerdefinierte Abfrage eingeben ausgewählt haben, geben Sie eine SQL-SELECT-Abfrage ein.
4. Geben Sie unter Benutzerdefinierte Teradata-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines Teradata-Zielknotens

Voraussetzungen

- Eine AWS Glue Teradata Vantage-Verbindung, konfiguriert mit einem AWS Secrets Manager Geheimnis, wie im vorherigen Abschnitt beschrieben,. the section called “Erstellen einer Teradata-Vantage-Verbindung”
- Entsprechende Berechtigungen für Ihren Auftrag zum Lesen des von der Verbindung verwendeten Secrets.
- Eine Teradata-Tabelle, in die Sie schreiben möchten,. *tableName*

Hinzufügen eines Teradata-Datenziels

Einen Knoten Datenziel – Teradata fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für die Teradata-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Teradata-Verbindung erstellen aus. Weitere Informationen finden Sie unter Übersicht über die Verwendung von Konnektoren und Verbindungen.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Konfigurieren Sie den Tabellennamen, indem Sie Folgendes angeben. *tableName*
3. Geben Sie unter Benutzerdefinierte Teradata-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines Teradata-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie bei der Programmierung AWS Glue für Spark-Skripte.

Siehe the section called “Teradata-Vantage-Verbindungen”.

Verbindung zu Twilio herstellen

Twilio bietet programmierbare Kommunikationstools zum Tätigen und Empfangen von Telefonanrufen, zum Senden und Empfangen von Textnachrichten sowie zum Ausführen anderer Kommunikationsfunktionen mithilfe seines Webdienstes. APIs Twilio APIs unterstützt seine Kommunikationsplattform. Dahinter verbirgt APIs sich eine Softwareschicht, die Kommunikationsnetzwerke auf der ganzen Welt verbindet und optimiert, sodass Ihre Benutzer weltweit jeden anrufen und Nachrichten senden können. Als Twilio-Benutzer können Sie eine Verbindung AWS Glue zu Ihrem Twilio-Konto herstellen. Anschließend können Sie Twilio als Datenquelle für Ihre ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Twilio und AWS Diensten oder anderen unterstützten Anwendungen zu übertragen.

Themen

- AWS Glue Unterstützung für Twilio
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Twilio konfigurieren

- Konfiguration von Twilio-Verbindungen
- Aus Twilio-Entitäten lesen
- Twilio-Verbindungsoptionen
- Einschränkungen und Hinweise für den Twilio Connector

AWS Glue Unterstützung für Twilio

AWS Glue unterstützt Twilio wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Twilio abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Twilio API-Versionen

Die folgenden Twilio-API-Versionen werden unterstützt:

- v1
- 2010-04-01

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
```

```
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Twilio konfigurieren

Bevor Sie Daten von Twilio AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Twilio-Konto mit Benutzername und Passwort.
- Ihr Twilio-Konto ist für den API-Zugriff aktiviert.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Twilio-Konto herstellen. Für typische Verbindungen müssen Sie in Twilio nichts weiter tun.

Konfiguration von Twilio-Verbindungen

Twilio unterstützt Benutzername und Passwort für die Standardauthentifizierung. Die Standardauthentifizierung ist eine einfache Authentifizierungsmethode, bei der Clients Anmeldeinformationen direkt eingeben, um auf geschützte Ressourcen zuzugreifen. AWS Glue kann den Benutzernamen (Konto-SID) und das Passwort (Auth Token) verwenden, um Twilio zu authentifizieren. APIs

Eine öffentliche Twilio-Dokumentation für den Ablauf der Standardauthentifizierung finden Sie unter Standardauthentifizierung | Twilio.

So konfigurieren Sie eine Twilio-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Für die Standardauthentifizierung: Das Secret sollte das Consumer Secret der verbundenen App mit der Konto-SID (Benutzername) und dem Auth-Token (Passwort) enthalten.

   > **Note**
   >
   > Sie müssen ein Geheimnis für Ihre Verbindungen in AWS Glue erstellen.

2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Twilio aus.
   b. Geben Sie die `Edge_Location` Twilio-Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
```

```
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Twilio-Entitäten lesen

Voraussetzung

Ein Twilio-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie `SMS-Message` oder`SMS-CountryPricing`.

Unterstützte Entitäten für die Quelle:

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
SMS-Nachricht	REST	Ja	Ja	Nein	Ja	Ja
SMS-CountryPricing	REST	Nein	Nein	Nein	Ja	Nein
Sprachanruf	REST	Ja	Ja	Nein	Ja	Nein

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Sprach-Anwendung	REST	Ja	Ja	Nein	Ja	Nein
Sprach-ID OutgoingCaller	REST	Ja	Ja	Nein	Ja	Nein
Sprachwarteschlange	REST	Ja	Ja	Nein	Ja	Nein
Konversationen — Konversation	REST	Ja	Ja	Nein	Ja	Nein
Konversationen — Benutzer	REST	Nein	Ja	Nein	Ja	Nein
Konversationen — Rolle	REST	Nein	Ja	Nein	Ja	Nein
Konversationen-Konfiguration	REST	Nein	Nein	Nein	Ja	Nein
Gespräche - AddressConfiguration	REST	Ja	Ja	Nein	Ja	Nein

Entität	Schnittstelle	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Gespräche - WebhookConfiguration	REST	Nein	Nein	Nein	Ja	Nein
Gespräche - ParticipantConversation	REST	Nein	Nein	Nein	Ja	Nein
Konversationen — Anmeldedaten	REST	Nein	Ja	Nein	Ja	Nein
Gespräche - ConversationService	REST	Nein	Ja	Nein	Ja	Nein

Beispiel:

```
twilio_read = glueContext.create_dynamic_frame.from_options(
    connection_type="twilio",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "sms-message",
        "API_VERSION": "2010-04-01",
        "Edge_Location": "sydney.us1"
    }
```

Angaben zur Twilio-Entität und zum Feld:

Entität	Feld	Datentyp	Unterstützte Operatoren
SMS-Nachricht	account_sid	String	N/A
	api_version	String	N/A
	body	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	gesendet_datum_	DateTime	>=, <=, =
	date_updated	DateTime	N/A
	Richtung	String	N/A
	error_code	Ganzzahl	N/A
	error_message	String	N/A
	from	Ganzzahl	=
	messaging_service_sid	String	N/A
	Anzahl_Medien	String	N/A
	Anzahl_Segmente	String	N/A
	price	String	N/A
	Preis/Einheit	Struct	N/A
	sid	Ganzzahl	N/A
	Status	String	N/A
	Unterressourcen-Uris	Zuordnung	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	to	Ganzzahl	=
	uri	DateTime	N/A
SMS- CountryPricing	country	String	N/A
	ISO_Land	String	N/A
	URL	String	N/A
	Preise für ausgehende SMS	Auflisten	N/A
	Preise für eingehende SMS	Auflisten	N/A
	Preis/Einheit	String	N/A
Sprachanruf	Konto_SID	String	N/A
	Anmerkung	String	N/A
	beantwortet_von	String	N/A
	api_version	String	N/A
	Name des Anrufers	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	Richtung	String	N/A
	duration	String	N/A
	end_time	DateTime	>=, <=, =

Entität	Feld	Datentyp	Unterstützte Operatoren
	weitergeleitet_von	String	N/A
	from	String	=
	von_formatiert	String	N/A
	group_sid	String	N/A
	Eltern_Anruf-SID	String	N/A
	telefonnummer_sid	String	N/A
	price	String	N/A
	Preis/Einheit	String	N/A
	sid	String	N/A
	start_time	DateTime	>=, <=, =
	Status	String	=
	Unterressourcen-Uris	String	N/A
	to	String	=
	to_formatiert	String	N/A
	Trunk_SID	String	N/A
	uri	String	N/A
	queue_time	String	N/A
Sprach-Anwendung	account_sid	String	N/A
	api_version	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	freundlicher_Name	String	=
	message_status_callback	String	N/A
	sid	String	N/A
	sms_fallback_method	String	N/A
	sms_fallback_url	String	N/A
	sms_Methode	String	N/A
	sms_status_callback	String	N/A
	sms_url	String	N/A
	status_callback	String	N/A
	status_callback_methode	String	N/A
	uri	String	N/A
	Suche nach der Sprachanrufer-ID	Boolesch	N/A
	voice_fallback_method	String	N/A
	voice_fallback_url	String	N/A
	voice_methode	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	voice_url	String	N/A
public_application_connect_enabled	Boolesch	N/A	
Sprach-ID OutgoingCaller	sid	String	N/A
	Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	account_sid	String	N/A
	freundlicher_Name	String	=
	phone_number	String	=
	uri	String	N/A
Sprachwarteschlange	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	aktuelle_Größe	Ganzzahl	N/A
	freundlicher_Name	String	N/A
	uri	String	N/A
	Kontonummer	String	N/A
	durchschnittliche Wartezeit	Ganzzahl	N/A
	sid	String	N/A
	max_size	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Konversationen — Konversation	account_sid	String	N/A
	chat_service_sid	String	N/A
	messaging_service_sid	String	N/A
	sid	String	N/A
	freundlicher_Name	String	N/A
	eindeutiger_Name	String	N/A
	Attribute	String	N/A
	state	String	=
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	Timer	Struct	N/A
	URL	String	N/A
	links	Struct	N/A
	Bindungen	Struct	N/A
	Startdatum	DateTime	=
	Enddatum	DateTime	=
	Zeitschaltuhren. DateInactive	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Zeitschaltuhren. DateClosed	String	N/A
Konversationen — Benutzer	sid	String	N/A
	account_sid	String	N/A
	chat_service_sid	String	N/A
	Rolle_SID	String	N/A
	Identitäts	String	N/A
	freundlicher_Name	String	N/A
	Attribute	String	N/A
	ist_online	Boolesch	N/A
	ist meldepflichtig	Boolesch	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	URL	String	N/A
	links	Struct	N/A
Konversationen — Rolle	sid	String	N/A
	account_sid	String	N/A
	chat_service_sid	String	N/A
	freundlicher_Name	String	N/A
	Typ	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	permissions	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	URL	String	N/A
Konversationen-Konfiguration	account_sid	Long	N/A
	default_chat_service_sid	String	N/A
	default_messaging_service_sid	String	N/A
	default_inactive_timer	String	N/A
	default_closed_timer	String	N/A
	URL	String	N/A
	links	Zuordnung	N/A
Gespräche-AddressConfiguration	sid	String	N/A
	account_sid	String	N/A
	Typ	String	N/A
	address	String	N/A
	freundlicher_Name	String	N/A
	automatische_Erstellung	Struct	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	URL	String	N/A
	adresse_land	String	N/A
	AutoCreation. Aktiviert	Boolesch	N/A
	AutoCreation.Typ	String	N/A
	AutoCreation.ConversationServiceSid	String	N/A
	AutoCreation.WebhookUrl	String	N/A
	AutoCreation.WebhookMethod	String	N/A
	AutoCreation.WebhookFilters	Auflisten	N/A
	AutoCreation.StudioFlowSid	String	N/A
	AutoCreation.StudioRetryCount	Ganzzahl	N/A
Gespräche-WebhookConfiguration	account_sid	String	N/A
	Methode	String	N/A
	-Filter	Auflisten	N/A
	pre_webhook_url	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	post_webhook-URL	String	N/A
	Ziel	String	N/A
	URL	String	N/A
Konversationen-ParticipantConversation	account_sid	String	N/A
	chat_service_sid	String	N/A
	Teilnehmer_SID	String	N/A
	Teilnehmer_Benutzer_SID	String	N/A
	Identität des Teilnehmers	String	N/A
	Mitteilnehmer_Nachrichtenübermittlung	Struct	N/A
	Konversations_SID	String	N/A
	Eindeutiger Name der Konversation	String	N/A
	Konversationsfreundlicher_Name	String	N/A
	Attribute für Konversationen	String	N/A
	Gesprächsdatum_erstellt	DateTime	N/A
	Gesprächsdatum_aktualisiert	DateTime	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Konversation_erstellt von	String	N/A
	Gesprächsstatus	String	N/A
	Gesprächstimer	Struct	N/A
	links	Zuordnung	N/A
	address	String	=
	Identitäts	String	=
Konversationsanmeldedaten	sid	String	N/A
	account_sid	String	N/A
	freundlicher_Name	String	N/A
	Typ	String	N/A
	Sandbox	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	dated_aktualisiert	DateTime	N/A
	URL	String	N/A
	Zertifikat	String	N/A
	private_key	String	N/A
	api_key	String	N/A
	Secret	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
Gespräche-ConversationService	sid	String	N/A
	account_sid	String	N/A
	freundlicher_Name	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	date_updated	DateTime	N/A
	URL	String	N/A
	links	Zuordnung	N/A

Abfragen partitionieren

Felder, die die Partitionierung unterstützen:

In Twilio unterstützen die DateTime Datentypfelder die feldbasierte Partitionierung.

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS falls Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

```
"2024-05-01T20:55:02.000Z"
```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
twilio_read = glueContext.create_dynamic_frame.from_options(
    connection_type="twilio",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "sms-message",
        "API_VERSION": "2010-04-01",
        "PARTITION_FIELD": "date_sent"
        "LOWER_BOUND": "2024-05-01T20:55:02.000Z"
        "UPPER_BOUND": "2024-06-01T20:55:02.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Twilio-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Twilio aufgeführt:

- ENTITY_NAME(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Twilio.
- EDGE_LOCATION(String) — (Erforderlich) Eine gültige Twilio-Edge-Position.
- API_VERSION(String) — (Erforderlich) Wird zum Lesen verwendet. Twilio Rest API-Version, die Sie verwenden möchten. Twilio unterstützt zwei API-Versionen: 'v1' und '2010-04-01'.
- SELECTED_FIELDS(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- FILTER_PREDICATE(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- QUERY(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- PARTITION_FIELD(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.
- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige Twilio-Instanz-URL.

Einschränkungen und Hinweise für den Twilio Connector

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Twilio-Connector:

- Die datensatzbasierte Partitionierung wird nicht unterstützt, da es keine Möglichkeit gibt, die Gesamtzahl der Datensätze aus Twilio abzurufen.
- Die Felder `date_sent`start_time, und `end_time` haben den Datentyp Datetime, aber beim Filtern unterstützen sie nur Datumswerte (Zeitkomponenten werden nicht berücksichtigt).
- Das Filtern der Felder „Von“ oder „Bis“ funktioniert nur, wenn die Werte kein Präfix enthalten (z. B. ein Protokoll oder eine Bezeichnung). Wenn ein Präfix vorhanden ist, funktioniert das Filtern nach dem entsprechenden Feld nicht. Wenn Sie beispielsweise „to“: „whatsapp: +14xxxxxxxxxx“ als Filter übergeben, gibt Twilio keine Antwort zurück. Du musst es als „an“: „+14xxxxxxxx“ übergeben, dann werden Datensätze zurückgegeben, sofern sie existieren.
- Der Feldfilter „Identität“ ist bei der Abfrage der Entität obligatorisch. `conversation-participant-conversation`

Verbindung zu Vertica herstellen in AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für Vertica. AWS Glue Studio bietet eine visuelle Oberfläche, über die Sie eine Verbindung zu Vertica herstellen, Datenintegrationsjobs erstellen und diese auf dem ausführen können AWS Glue Studio serverlose Spark-Laufzeit.

AWS Glue Studio erstellt eine einheitliche Verbindung für Vertica. Weitere Informationen finden Sie unter Überlegungen.

Themen

- Erstellen einer Vertica-Verbindung
- Erstellen eines Vertica-Quellknotens

- Erstellen eines Vertica-Zielknotens
- Erweiterte Optionen

Erstellen einer Vertica-Verbindung

Voraussetzungen:

- Ein Amazon S3 S3-Bucket oder -Ordner, der als temporärer Speicher verwendet wird, wenn aus der Datenbank gelesen und in die Datenbank geschrieben wird, auf den von`tempS3Path`.

> **Note**
>
> Wenn Sie Vertica in der Vorschau von AWS Glue Auftragsdaten verwenden, werden temporäre Dateien möglicherweise nicht automatisch entfernt. `tempS3Path` Um sicherzustellen, dass temporäre Dateien entfernt werden, beenden Sie die Datenvorschau-Sitzung direkt, indem Sie im Bereich Datenvorschau die Option Sitzung beenden wählen. Wenn Sie nicht garantieren können, dass die Datenvorschau-Sitzung direkt beendet wird, sollten Sie die Amazon-S3-Lifecycle-Konfiguration so einrichten, dass alte Daten entfernt werden. Wir empfehlen, Daten zu entfernen, die älter als 49 Stunden sind, basierend auf der maximalen Auftragslaufzeit zuzüglich einer Marge. Weitere Informationen zur Konfiguration des Amazon-S3-Lebenszyklus finden Sie in der Amazon-S3-Dokumentation unter Verwalten Ihres Speicherlebenszyklus.

- Eine IAM-Richtlinie mit entsprechenden Berechtigungen für Ihren Amazon S3 S3-Pfad, die Sie Ihrer AWS Glue beruflichen Rolle zuordnen können.
- Wenn sich Ihre Vertica-Instance in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue Job mit der Vertica-Instance kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die bei der Ausführung des Jobs verwendet AWS Glue werden. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer Vertica-Instance und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem Vertica-Client-Port (Standard 5433) herstellen. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

Eine Verbindung zu Vertica konfigurieren Sie wie folgt:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer Vertica-Anmeldeinformationen einen geheimen Schlüssel, und. *verticaUsername* *verticaPassword* Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *verticaUsername*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *verticaPassword*

2. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter folgen. the section called “AWS Glue Verbindung hinzufügen” Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *connectionName*

   - Wählen Sie als Verbindungstyp Vertica aus.
   - Geben Sie als Vertica-Host den Hostnamen Ihrer Vertica-Installation an.
   - Geben Sie als Vertica-Port den Port an, über den Ihre Vertica-Installation verfügbar ist.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*secretName*.

3. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für Vertica-Instances, die AWS in einer Amazon VPC gehostet werden
     - Geben Sie Amazon VPC-Verbindungsinformationen für die Verbindung ein, die Ihre AWS Glue Vertica-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Sie müssen die folgenden Schritte ausführen, bevor Sie Ihren Job ausführen können AWS Glue :

- Erteilen Sie die mit Ihrem AWS Glue Job verknüpfte IAM-Rolle auf*tempS3Path*.
- Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle die Leseberechtigung. *secretName*

Erstellen eines Vertica-Quellknotens

Voraussetzungen

- Eine AWS Glue Datenkatalogverbindung vom Typ Vertica *connectionName* und ein temporärer Amazon S3 S3-Standort*tempS3Path*, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Vertica-Verbindung”.
- Eine Vertica-Tabelle, aus der Sie lesen*tableName*, oder abfragen möchten. *targetQuery*

Hinzufügen einer Vertica-Datenquelle

Einen Knoten Datenquelle – Vertica fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Vertica-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Vertica-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Vertica-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie die Datenbank aus, die Ihre Tabelle enthält.
3. Wählen Sie den Staging-Bereich in Amazon S3 und geben Sie eine S3A-URI für ein. *tempS3Path*
4. Wählen Sie die Vertica-Quelle aus.

   - Eine einzelne Tabelle auswählen – rufen Sie alle Daten aus einer einzigen Tabelle ab.
   - Benutzerdefinierte Abfrage eingeben – rufen Sie einen Datensatz aus mehreren Tabellen ab, basierend auf Ihrer benutzerdefinierten Abfrage.
5. Wenn Sie eine einzelne Tabelle ausgewählt haben, geben Sie ein Schema ein ***tableName*** und wählen Sie es optional aus.

   Wenn Sie Benutzerdefinierte Abfrage eingeben ausgewählt haben, geben Sie eine SQL-SELECT-Abfrage ein und wählen Sie optional ein Schema aus.
6. Geben Sie unter Benutzerdefinierte Vertica-Eigenschaften nach Bedarf Parameter und Werte ein.

Erstellen eines Vertica-Zielknotens

Voraussetzungen

- Eine AWS Glue Datenkatalogverbindung vom Typ Vertica *connectionName* und ein temporärer Amazon S3 S3-Standort*tempS3Path*, wie im vorherigen Abschnitt beschrieben,the section called “Erstellen einer Vertica-Verbindung”.

Hinzufügen eines Vertica-Datenziels

Einen Knoten Datenziel – Vertica fügen Sie wie folgt hinzu:

1. Wählen Sie die Verbindung für Ihre Vertica-Datenquelle aus. Da Sie sie erstellt haben, sollte sie in der Dropdown-Liste verfügbar sein. Wenn Sie eine Verbindung erstellen müssen, wählen Sie Vertica-Verbindung erstellen aus. Weitere Informationen finden Sie im vorhergehenden Abschnitt the section called “Erstellen einer Vertica-Verbindung”.

   Nachdem Sie eine Verbindung ausgewählt haben, können Sie die Verbindungseigenschaften anzeigen, indem Sie auf Eigenschaften anzeigen klicken.
2. Wählen Sie die Datenbank aus, die Ihre Tabelle enthält.
3. Wählen Sie den Staging-Bereich in Amazon S3 und geben Sie eine S3A-URI für ein. *tempS3Path*
4. Geben Sie ein ***tableName*** Schema ein und wählen Sie es optional aus.
5. Geben Sie unter Benutzerdefinierte Vertica-Eigenschaften nach Bedarf Parameter und Werte ein.

Erweiterte Optionen

Sie können bei der Erstellung eines Vertica-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie die, die bei der Programmierung AWS Glue für Spark-Skripte verfügbar sind.

Siehe the section called “Vertica-Verbindungen”.

Verbindung herstellen zu WooCommerce

WooCommerce ist eine flexible Open-Source-Softwarelösung, die für WordPress basierte Websites entwickelt wurde. Es wird häufig zur Erstellung von Online-E-Commerce-Shops verwendet. Mit

dieser Softwarelösung kann jeder seine reguläre Website in einen voll funktionsfähigen Online-Shop verwandeln.

Themen

- AWS Glue Unterstützung für WooCommerce
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration WooCommerce
- WooCommerce Verbindungen konfigurieren
- Aus WooCommerce Entitäten lesen
- WooCommerce Verbindungsoptionen

AWS Glue Unterstützung für WooCommerce

AWS Glue unterstützt WooCommerce wie folgt:

Als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von abzufragen WooCommerce.

Wird es als Ziel unterstützt?

Nein.

Unterstützte WooCommerce API-Versionen

Die folgenden WooCommerce API-Versionen werden unterstützt:

- v3

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
 }
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration WooCommerce

Bevor Sie Daten von AWS Glue übertragen können WooCommerce, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein WooCommerce Konto mit a `consumerKey` und `consumerSecret` a.
- Ihr WooCommerce Konto hat API-Zugriff mit einer gültigen Lizenz.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem WooCommerce Konto herstellen. Für typische Verbindungen müssen Sie nichts weiter tun WooCommerce.

WooCommerce Verbindungen konfigurieren

WooCommerce unterstützt benutzerdefinierte Authentifizierung. Öffentliche WooCommerce Dokumentation zur Generierung der erforderlichen API-Schlüssel für die benutzerdefinierte Autorisierung finden Sie unter Authentifizierung — WooCommerce REST-API-Dokumentation.

So konfigurieren Sie eine WooCommerce Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   - Bei einer vom Kunden verwalteten verbundenen App sollte das Secret das Consumer Secret der verbundenen App mit `consumerKey` und `consumerSecret` als Schlüssel enthalten. Hinweis: Sie müssen pro Verbindung in ein Secret erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps WooCommerce.
   b. Geben Sie die WooCommerce Instanz `INSTANCE_URL` an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus WooCommerce Entitäten lesen

Voraussetzung

Ein WooCommerce Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Gutschein, Bestellung, Produkt usw.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Coupon	Ja	Ja	Ja	Ja	Ja
Gutschein insgesamt	Nein	Nein	Nein	Ja	Nein
Kunden insgesamt	Nein	Nein	Nein	Ja	Nein
Order	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Bestellungen insgesamt	Nein	Nein	Nein	Ja	Nein
Zahlungsgateway	Nein	Nein	Nein	Ja	Nein
Produkt	Ja	Ja	Ja	Ja	Ja
Produktattribut	Ja	Ja	Ja	Ja	Ja
Kategorie des Produkts	Ja	Ja	Ja	Ja	Ja
Bewertung des Produkts	Ja	Ja	Ja	Ja	Ja
Versandklasse des Produkts	Ja	Ja	Ja	Ja	Ja
Produktetikett	Ja	Ja	Ja	Ja	Ja
Produktvariante	Ja	Ja	Ja	Ja	Ja
Produkte insgesamt	Nein	Nein	Nein	Ja	Nein
Bericht (Liste)	Nein	Nein	Nein	Ja	Nein
Bewertungen insgesamt	Nein	Nein	Nein	Ja	Nein
Verkaufsbericht	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Versandart	Nein	Nein	Nein	Ja	Nein
Versandzone	Nein	Nein	Nein	Ja	Nein
Standort der Versandzone	Nein	Nein	Nein	Ja	Nein
Methode der Versandzone	Nein	Nein	Nein	Ja	Nein
Steuersatz	Ja	Ja	Ja	Ja	Ja
Steuerklasse	Nein	Nein	Nein	Ja	Nein
Bericht über Topseller	Ja	Nein	Nein	Ja	Nein

Beispiel:

```
woocommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.woocommerce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupon",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
    }
```

WooCommerce Entitäts- und Felddetails:

Entität	Feld	Datentyp	Unterstützte Operatoren
Coupon	id	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Code	String	EQUAL_TO
	Betrag	String	N/A
	Status	String	N/A
	date_created	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	Art des Rabatts	String	N/A
	description	String	N/A
	date_läuft ab	String	N/A
	date_expires_gmt	String	N/A
	Anzahl der Nutzungen	Ganzzahl	N/A
	individuelle_Nutzung	Boolesch	N/A
	Produktkennungen	Auflisten	N/A
	ausgeschlossene_Produktkennungen	Auflisten	N/A
	usage_limit	Ganzzahl	N/A
	Nutzungsbeschränkung pro Benutzer	Ganzzahl	N/A
	Beschränken Sie die Nutzung auf_x_Artikel	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	kostenloser Versand	Boolesch	N/A
	produktkategorien	Auflisten	N/A
	ausgeschlossene_Produktkategorien	Auflisten	N/A
	Verkaufsartikel ausschließen	Boolesch	N/A
	Mindestbetrag	String	N/A
	maximaler_Betrag	String	N/A
	E-Mail-Einschränkungen	Auflisten	N/A
	verwendet_von	Auflisten	N/A
	meta_data	Auflisten	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	nach	DateTime	EQUAL_TO
	vor	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	geändert_nachher	DateTime	EQUAL_TO
	vorher geändert_	DateTime	EQUAL_TO
	Datume_are_gmt	Boolesch	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
Summe des Gutscheins	Schnecke	String	N/A
	Name	String	N/A
	total	Ganzzahl	N/A
Gesamtzahl der Kunden	Schnecke	String	N/A
	Name	String	N/A
	total	Ganzzahl	N/A
order	id	Ganzzahl	N/A
	Eltern-ID	Ganzzahl	N/A
	Zahl	String	N/A
	Bestellschlüssel	String	N/A
	erstellt_über	String	N/A
	Status	String	N/A
	currency	String	N/A
	version	String	N/A
	Erstellungsdatum/Erstellungsdatum	DateTime	N/A
	Datum_geändert	DateTime	N/A
	rabatt_total	String	N/A
	rabatt_tax	String	N/A
	Gesamtbetrag der Versandkosten	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Versandsteuer	String	N/A
	Warenkorbsteuer	String	N/A
	total	String	N/A
	Gesamtzahl der Steuern	String	N/A
	Die Preise beinhalten Steuern	Boolesch	N/A
	customer_id	Ganzzahl	N/A
	IP-Adresse des Kunden	String	N/A
	kunden_user_agent	String	N/A
	Anmerkung des Kunden	String	N/A
	Fakturierung	Struct	N/A
	Versand	Struct	N/A
	Zahlungsmethode	String	N/A
	Titel der Zahlungsmethode	String	N/A
	transaction_id	String	N/A
	Datum_bezahlt	DateTime	N/A
	datum_abgeschlossen	DateTime	N/A
	cart_hash	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	meta_data	Auflisten	N/A
	Zeilenelemente	Auflisten	N/A
	Steuerlinien	Auflisten	N/A
	Versandlinien	Auflisten	N/A
	gebührenpflichtige Linien	Auflisten	N/A
	Couponlines	Auflisten	N/A
	Rückerstattungen	Auflisten	N/A
	Zahlungs-URL	String	N/A
	ist_editierbar	Boolesch	N/A
	benötigte_Zahlung	Boolesch	N/A
	benötigt Bearbeitung	Boolesch	N/A
	Erstellungsdatum_GMT	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	date_completed_gmt	DateTime	N/A
	date_paid_gmt	DateTime	N/A
	Währungssymbol	String	N/A
	set_paid	Boolesch	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	nach	DateTime	EQUAL_TO
	vor	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	customer	Ganzzahl	EQUAL_TO
	Produkt	Ganzzahl	EQUAL_TO
	dp	Ganzzahl	EQUAL_TO
	vorher geändert_	DateTime	EQUAL_TO
	geändert_nachher	DateTime	EQUAL_TO
	Datumsangaben sind GMT	Boolesch	EQUAL_TO
Summe der Bestellungen	Schnecke	String	N/A
	Name	String	N/A
	total	Ganzzahl	N/A
Zahlungsgateway	Titel	String	N/A
	description	String	N/A
	order	String	N/A
	aktiviert	Boolesch	N/A
	Methodentitel	String	N/A
	Methodenbeschreibung	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	method_supports	Auflisten	N/A
	Einstellungen	String	N/A
	benötigte_Einrichtung	Boolesch	N/A
	Skripte nach der Installation	Auflisten	N/A
	Einstellungen_URL	String	N/A
	Verbindungs-URL	String	N/A
	setup_hilfetext	String	N/A
	erforderliche_Einstellungsschlüssel	Auflisten	N/A
Produkt	id	Ganzzahl	N/A
	Name	String	N/A
	Typ	String	EQUAL_TO
	Permalink	String	N/A
	Erstellungsdatum	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	Sichtbarkeit des Katalogs	String	N/A
	description	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	short_description	String	N/A
	price	String	N/A
	regulärer Preis	String	N/A
	verkaufspreis	String	N/A
	Datum des Verkaufs von	DateTime	N/A
	Verkaufsdatum ab GMT	DateTime	N/A
	Verkaufsdatum_bis	DateTime	N/A
	Datum des Verkaufs bis GMT	DateTime	N/A
	preis_html	String	N/A
	käuflich	Boolesch	N/A
	Gesamtumsatz	Ganzzahl	N/A
	virtuell	Boolesch	N/A
	herunterladbar	Boolesch	N/A
	Downloads	Auflisten	N/A
	herunterladen_limit	Ganzzahl	N/A
	Ablauf des Herunterladens	Ganzzahl	N/A
	externe_URL	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Schaltflächentext	String	N/A
	Steuerstatus	String	N/A
	Bestand verwalten	Boolesch	N/A
	Lagerbestandsmenge	Ganzzahl	N/A
	Nachbestellungen	String	N/A
	Backorders_erlaubt	Boolesch	N/A
	nachbestellt	Boolesch	N/A
	wird einzeln verkauft	Boolesch	N/A
	Gewicht	String	N/A
	Dimensionen	Struct	N/A
	Versand_erforderlich	Boolesch	N/A
	versand_steuerpflichtig	Boolesch	N/A
	Versandklassen-ID	Ganzzahl	N/A
	Bewertungen_erlaubt	Boolesch	N/A
	durchschnittliche Bewertung	String	N/A
	Bewertung_Anzahl	Ganzzahl	N/A
	verwandte_IDs	Auflisten	N/A
	Upselling-IDs	Auflisten	N/A
	Cross-Sell-IDs	Auflisten	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Eltern-ID	Ganzzahl	N/A
	Kaufhinweis	String	N/A
	categories	Auflisten	N/A
	tags	Auflisten	N/A
	Bilder	Auflisten	N/A
	Attribute	Auflisten	N/A
	default_attributes	Auflisten	N/A
	Variationen	Auflisten	N/A
	gruppierte_Produkte	Auflisten	N/A
	menü_reihenfolge	Ganzzahl	N/A
	meta_data	Auflisten	N/A
	niedriger_Lagerbestand	Ganzzahl	N/A
	jetpack veröffentlicht Verbindungen	Auflisten	N/A
	jetpack-related-posts	Auflisten	N/A
	jetpack_likes_aktiviert	Boolesch	N/A
	jetpack_sharing_aktiviert	Boolesch	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	nach	DateTime	EQUAL_TO
	vor	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	Schnecke	String	EQUAL_TO
	Status	String	EQUAL_TO
	sku	String	EQUAL_TO
	aufgeführt	Boolesch	EQUAL_TO
	Tag (Markierung)	String	EQUAL_TO
	Versandklasse	String	EQUAL_TO
	Steuerklasse	String	EQUAL_TO
	zum Verkauf	Boolesch	EQUAL_TO
	stock_status	String	EQUAL_TO
	hat_Optionen	Boolesch	N/A
	nachher geändert	DateTime	EQUAL_TO
	vorher geändert_	DateTime	EQUAL_TO
	Datume_are_gmt	Boolesch	EQUAL_TO
	Kategorie	String	EQUAL_TO
	Attribut	String	EQUAL_TO
	mindest_preis	String	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	max_price	String	EQUAL_TO
Produktattribut	id	Ganzzahl	N/A
	Name	String	N/A
	Schnecke	String	N/A
	Typ	String	N/A
	order_by	String	N/A
	hat_archive	Boolesch	N/A
	context	String	EQUAL_TO
product-attribute-term	id	Ganzzahl	N/A
	Name	String	N/A
	Schnecke	String	N/A
	description	String	N/A
	Menüreihenfolge	Ganzzahl	N/A
	count	Ganzzahl	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	hide_empty	Boolesch	EQUAL_TO
	übergeordneter	Ganzzahl	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	Produkt	Ganzzahl	EQUAL_TO
Produktkategorie	id	Ganzzahl	N/A
	Name	String	N/A
	Schnecke	String	EQUAL_TO
	description	String	N/A
	display	String	N/A
	Abbild	Struct	N/A
	Menüreihenfolge	Ganzzahl	N/A
	count	Ganzzahl	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	hide_empty	Boolesch	EQUAL_TO
	übergeordneter	Ganzzahl	EQUAL_TO
	Produkt	Ganzzahl	EQUAL_TO
Produktbewertung	id	Ganzzahl	N/A
	Erstellungsdatum	DateTime	N/A
	date_created_gmt	DateTime	N/A
	product_id	Ganzzahl	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	product_name	String	N/A
	produkt_permalink	String	N/A
	Rezension	String	N/A
	Bewertung	Ganzzahl	N/A
	verified	Boolesch	N/A
	Prüfer	String	N/A
	E-Mail des Rezensenten	String	N/A
	Avatar-URLs des Rezensenten	Struct	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	nach	DateTime	EQUAL_TO
	vor	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	Status	String	EQUAL_TO
product-shipping-class	id	Ganzzahl	N/A
	Name	String	N/A
	Schnecke	String	EQUAL_TO
	description	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	count	Ganzzahl	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	hide_empty	String	EQUAL_TO
	Produkt	Ganzzahl	EQUAL_TO
Produkt-Tag	id	Ganzzahl	N/A
	Name	String	N/A
	Schnecke	String	EQUAL_TO
	description	String	N/A
	count	Ganzzahl	N/A
	context	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	hide_empty	Boolesch	EQUAL_TO
	Produkt	Ganzzahl	EQUAL_TO
Produkt insgesamt	Schnecke	String	N/A
	Name	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	total	Ganzzahl	N/A
Produktvariation	id	Ganzzahl	N/A
	Erstellungsdatum	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	description	String	N/A
	Permalink	String	N/A
	price	String	N/A
	regulärer Preis	String	N/A
	verkaufspreis	String	N/A
	Datum des Verkaufs von	DateTime	N/A
	Verkaufsdatum ab GMT	DateTime	N/A
	Verkaufsdatum_bis	DateTime	N/A
	Datum des Verkaufs bis GMT	DateTime	N/A
	käuflich	Boolesch	N/A
	virtuell	Boolesch	N/A
	herunterladbar	Boolesch	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	Downloads	Auflisten	N/A
	herunterladen_limit	Ganzzahl	N/A
	Ablauf des Herunterladens	Ganzzahl	N/A
	Steuerstatus	String	N/A
	Bestand verwalten	Boolesch	N/A
	Lagerbestandsmenge	Ganzzahl	N/A
	Nachbestellungen	String	N/A
	Backorders_erlaubt	Boolesch	N/A
	nachbestellt	Boolesch	N/A
	niedriger Lagerbestand	Ganzzahl	N/A
	Gewicht	String	N/A
	Dimensionen	Struct	N/A
	Versandklasse	String	N/A
	Versandklassen-ID	Ganzzahl	N/A
	Abbild	Struct	N/A
	Attribute	Auflisten	N/A
	menü_bestellung	Ganzzahl	N/A
	meta_data	Auflisten	N/A
	context	String	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	search	String	EQUAL_TO
	nach	DateTime	EQUAL_TO
	vor	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	ordnen nach	String	EQUAL_TO
	Schnecke	String	EQUAL_TO
	Status	String	EQUAL_TO
	sku	String	EQUAL_TO
	tax_class	String	EQUAL_TO
	zum Verkauf	Boolesch	EQUAL_TO
	Mindestpreis	String	EQUAL_TO
	max_price	String	EQUAL_TO
	stock_status	String	EQUAL_TO
report	Schnecke	String	N/A
	description	String	N/A
Bewertung insgesamt	Schnecke	String	N/A
	Name	String	N/A
	total	Ganzzahl	N/A
Verkaufsbericht	Gesamtumsatz	String	N/A
	Nettoumsatz	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	durchschnittlicher Umsatz	String	N/A
	Bestellungen insgesamt	Ganzzahl	N/A
	Gesamtzahl der Artikel	Ganzzahl	N/A
	Steuern insgesamt	String	N/A
	Versand insgesamt	String	N/A
	gesamte_Rückerstattungen	Ganzzahl	N/A
	total_rabatt	String	N/A
	Gesamtsummen_gruppiert nach	String	N/A
	Summen	Struct	N/A
	Gesamtzahl der Kunden	Ganzzahl	N/A
	context	String	EQUAL_TO
	Zeitraum	String	EQUAL_TO
	Datum_min	Datum	EQUAL_TO
	Datum_max	Datum	EQUAL_TO
Versandart	id	String	N/A
	Titel	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	description	String	N/A
Versandzone	id	Ganzzahl	EQUAL_TO
	Name	String	N/A
	order	Ganzzahl	N/A
shipping-zone-location	Code	String	N/A
	Typ	String	N/A
shipping-zone-method	instance_id	Ganzzahl	N/A
	id	Ganzzahl	EQUAL_TO
	Titel	String	N/A
	order	Ganzzahl	N/A
	aktiviert	Boolesch	N/A
	methoden_id	String	N/A
	Methodentitel	String	N/A
	Methodenbeschreibung	String	N/A
	Einstellungen	Struct	N/A
Steuerklasse	Schnecke	String	N/A
	Name	String	N/A
Steuersatz	id	Ganzzahl	N/A
	country	String	N/A

Entität	Feld	Datentyp	Unterstützte Operatoren
	state	String	N/A
	plz	String	N/A
	city	String	N/A
	Postleitzahlen	Auflisten	N/A
	Städte	Auflisten	N/A
	bewerten	String	N/A
	Name	String	N/A
	priority	Ganzzahl	N/A
	Verbindung	Boolesch	N/A
	Versand	Boolesch	N/A
	context	String	EQUAL_TO
	order	String	EQUAL_TO
	bestellen von	String	EQUAL_TO
	class	String	EQUAL_TO
top-seller-report	Name	String	N/A
	product_id	Ganzzahl	N/A
	quantity	Ganzzahl	N/A
	context	String	EQUAL_TO
	Zeitraum	String	EQUAL_TO
	date_min	Datum	EQUAL_TO

Entität	Feld	Datentyp	Unterstützte Operatoren
	Datum_max	Datum	EQUAL_TO

> **Note**
>
> Die Datentypen Struct und List werden in den Datentyp String konvertiert, und der DateTime Datentyp wird in der Antwort der Konnektoren in Timestamp konvertiert.

Abfragen partitionieren

Datensatzbasierte Partitionierung:

Sie können die zusätzliche Spark-Option bereitstellen, `NUM_PARTITIONS` wenn Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine `NUM_PARTITIONS` Anzahl von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

Bei der datensatzbasierten Partitionierung wird die Gesamtzahl der vorhandenen Datensätze von der WooCommerce API abgefragt und durch eine angegebene Zahl geteilt. `NUM_PARTITIONS` Die resultierende Anzahl von Datensätzen wird dann gleichzeitig von jeder Unterabfrage abgerufen.

- `NUM_PARTITIONS`: die Anzahl der Partitionen.

Die folgenden Entitäten unterstützen die datensatzbasierte Partitionierung:

- Coupon
- order
- Produkt
- Produktattribut
- product-attribute-term
- Produktkategorie
- Produktbewertung
- product-shipping-class

- Produktetikett
- Produktvariante
- Steuersatz

Beispiel:

```
woocommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.woocommerce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupon",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
        "NUM_PARTITIONS": "10"
    }
```

Datensatzbasierte Partitionierung:

Die ursprüngliche Abfrage ist in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können:

- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
WooCommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="WooCommerce",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "1234567890123456789",
        "ENTITY_NAME": "Bill",
        "API_VERSION": "v3",
        "NUM_PARTITIONS": "10"
    }
```

WooCommerce Verbindungsoptionen

Im Folgenden finden Sie Verbindungsoptionen für WooCommerce:

- `ENTITY_NAME`(String) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in WooCommerce.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. WooCommerce Rest-API-Version, die Sie verwenden möchten.
- `REALM_ID`(Zeichenfolge) — Eine ID, die ein einzelnes WooCommerce Online-Unternehmen identifiziert, an das Sie Anfragen senden.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `INSTANCE_URL`(String) — (Erforderlich) Eine gültige WooCommerce Instanz-URL im Format: https://<instance>.wpcomstaging.com
- `NUM_PARTITIONS`(Integer) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

Verbindung zu Zendesk herstellen

Zendesk ist eine cloudbasierte Helpdesk-Managementlösung, die anpassbare Tools zum Aufbau eines Kundendienstportals, einer Wissensdatenbank und Online-Communities bietet.

Themen

- AWS Glue Unterstützung für Zendesk
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Zendesk
- Konfiguration von Zendesk-Verbindungen
- Aus Zendesk-Entitäten lesen
- Zendesk-Verbindungsoptionen
- Einschränkungen

AWS Glue Unterstützung für Zendesk

AWS Glue unterstützt Zendesk wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten von Zendesk abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte Zendesk-API-Versionen

Die folgenden Zendesk-API-Versionen werden unterstützt

- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, der die Richtlinie zugeordnet ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird normalerweise Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Zendesk

Bevor Sie Daten von Zendesk AWS Glue übertragen können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Zendesk-Konto. Weitere Informationen finden Sie unter Ein Zendesk-Konto erstellen.
- Ihr Zendesk-Konto ist für den API-Zugriff aktiviert.
- Ihr Zendesk-Konto ermöglicht es Ihnen, verbundene Apps zu installieren.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Zendesk-Konto herstellen.

Ein Zendesk-Konto erstellen

Um ein Zendesk-Konto zu erstellen:

1. Gehen Sie zur https://www.zendesk.com/in/ Registrierung/
2. Geben Sie die Details wie Ihre geschäftliche E-Mail-Adresse, Vorname, Nachname, Telefonnummer, Berufsbezeichnung, Firmenname, Anzahl der Mitarbeiter im Unternehmen, Passwort und bevorzugte Sprache ein. Wählen Sie dann Registrierung für die Testversion abschließen.

3. Sobald Ihr Konto erstellt wurde, füllen Sie den Bestätigungslink aus, den Sie erhalten haben, um Ihre E-Mail-Adresse zu verifizieren.
4. Sobald die geschäftliche E-Mail-Adresse verifiziert ist, werden Sie zu Ihrem Zendesk-Konto weitergeleitet. Wählen Sie die Option Zendesk kaufen für Ihren bevorzugten Plan. Hinweis: Für den Zendesk Connector wird empfohlen, den Suite Enterprise-Plan zu erwerben.

Eine Client-App und OAuth 2.0-Anmeldeinformationen erstellen

So erstellen Sie eine Client-App und OAuth 2.0-Anmeldeinformationen:

1. Melden Sie sich bei Ihrem Zendesk-Konto an, in dem die OAuth 2.0-App erstellt https://www.zendesk.com/in/ werden soll login/
2. Klicken Sie auf das Zahnradsymbol. Wählen Sie den Link Gehe zum Admin Center, um die Admin Center-Seite zu öffnen.
3. Wählen Sie in der linken Seitenleiste Apps und Integrationen und dann APIs> Zendesk-API.
4. Wählen Sie auf der Zendesk-API-Seite den Tab Kunden aus. OAuth
5. Wählen Sie auf der rechten Seite Oauth-Client hinzufügen aus.
6. Füllen Sie die folgenden Felder aus, um einen Client zu erstellen:
   a. Kundenname — Geben Sie einen Namen für Ihre App ein. Dies ist der Name, den Benutzer sehen, wenn sie aufgefordert werden, Zugriff auf Ihre Anwendung zu gewähren, und wenn sie die Liste der Apps von Drittanbietern überprüfen, die Zugriff auf ihren Zendesk haben.
   b. Beschreibung — Fakultativ. Eine kurze Beschreibung Ihrer App, die Benutzern angezeigt wird, wenn sie aufgefordert werden, Zugriff darauf zu gewähren.
   c. Firma — Fakultativ. Der Firmenname, den Benutzer sehen, wenn sie aufgefordert werden, Zugriff auf Ihre Anwendung zu gewähren. Anhand dieser Informationen können sie besser verstehen, wem sie Zugriff gewähren.
   d. Logo — Fakultativ. Dies ist das Logo, das Benutzern angezeigt wird, wenn sie aufgefordert werden, Zugriff auf Ihre Anwendung zu gewähren. Das Bild kann JPG, GIF oder PNG sein. Die besten Ergebnisse erzielen Sie, wenn Sie ein quadratisches Bild hochladen. Die Größe wird für die Autorisierungsseite angepasst.
   e. Eindeutiger Bezeichner — Das Feld wird automatisch mit einer neu formatierten Version des Namens gefüllt, den Sie für Ihre App eingegeben haben. Sie können ihn auf Wunsch ändern.
   f. Weiterleitung URLs — Geben Sie die URL ein URLs , die Zendesk verwenden soll, um die Entscheidung des Benutzers zu senden, Zugriff auf Ihre Anwendung zu gewähren.

Zum Beispiel: oauth https://us-east-1.console.aws.amazon.com/gluestudio/

7. Klicken Sie auf Speichern.
8. Nach der Aktualisierung der Seite erscheint auf der unteren Seite ein neues, vorab ausgefülltes geheimes Feld. Dies ist der in der Spezifikation angegebene Wert für „client_secret“. OAuth2 Kopieren Sie den Secret-Wert in Ihre Zwischenablage und speichern Sie ihn an einem sicheren Ort. Hinweis: Die Zeichen können die Breite des Textfeldes überschreiten. Achten Sie daher darauf, vor dem Kopieren alles auszuwählen.
9. Klicken Sie auf Speichern.

Konfiguration von Zendesk-Verbindungen

Der Zendesk-Connector unterstützt den Gewährungstyp Autorisierungscode.

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, kann sich standardmäßig auf eine AWS Glue eigene verbundene App (AWS Glue verwaltete Client-Anwendung) verlassen, für die er außer der URL seiner Zendesk-Instanz keine OAuth weiteren Informationen angeben muss. Die AWS Glue Konsole leitet den Benutzer zu Zendesk weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Zendesk-Instanz gewähren muss.
- Sie können sich dennoch dafür entscheiden, Ihre eigene verbundene App in Zendesk zu erstellen und beim Herstellen von Verbindungen über die Konsole Ihre eigene Kunden-ID und Ihr eigenes Kundengeheimnis anzugeben. AWS Glue In diesem Szenario werden Sie weiterhin zu Zendesk weitergeleitet, um sich anzumelden und den Zugriff auf Ihre Ressourcen AWS Glue zu autorisieren.
- Diese Art der Gewährung führt zu einem Zugriffstoken. Das Zugriffstoken läuft nie ab.

Eine öffentliche Zendesk-Dokumentation zur Erstellung einer verbundenen App für den Authorization OAuth Code-Flow finden Sie unter OAuth Tokens for Grant Types.

So konfigurieren Sie eine Zendesk-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:

a. Für den AuthorizationCode Gewährungstyp: Für eine vom Kunden verwaltete verbundene App sollte das Geheimnis das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.

b. Hinweis: Sie müssen ein Geheimnis pro Verbindung in erstellen AWS Glue.

2. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:

a. Wählen Sie bei der Auswahl eines Verbindungstyps Zendesk aus.

b. Geben Sie die INSTANCE_URL des Zendesk an, zu dem Sie eine Verbindung herstellen möchten.

c. Geben Sie die Zendesk-Umgebung an.

d. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

e. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

f. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

3. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Zendesk-Entitäten lesen

Voraussetzung

Ein Zendesk-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie Ticket, Benutzer oder Artikel, wie in der folgenden Tabelle angegeben.

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Fahrkarte	Y	Y	Y	Y	N
Benutzer	Y	Y	Y	Y	N
Organisation	Y	Y	Y	Y	N
Artikel	Y	Y	N	Y	N
Veranstaltung einlösen	Y	Y	N	Y	N
Event Metric Ticket	Y	Y	N	Y	N
Kommentar zum Ticket	Y	Y	Y	Y	N
Ticket-Feld	Y	Y	N	Y	N
Ticket-Metrik	Y	Y	N	Y	N
Ticket-Aktivität	Y	Y	N	Y	N
Ticket überspringen	N	Y	N	Y	N
Gruppe	Y	Y	Y	Y	N

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Mitgliedschaft in einer Gruppe	N	Y	Y	Y	N
Zufriedenheitsbewertung	Y	Y	N	Y	N
Anzeigen	Y	Y	Y	Y	N
Auslöser	Y	Y	Y	Y	N
Kategorie auslösen	N	Y	Y	Y	N
Makro	Y	Y	Y	Y	N
Automatisierung	N	Y	Y	Y	N

Beispiel:

```
Zendesk_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Zendesk",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v2"
    }
```

Zendesk-Entitäten und Felddetails:

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
Artikel	URL	String		
	id	Long		
	author_id	Long		
	body	String		
	Kommentare_deaktiviert	Boolesch		
	Einziehung	Boolesch		
	bearbeitet_am	DateTime		
	html_URL	String		
	Labelnamen	Auflisten		
	locale	String	EQUAL_TO	
	veraltet	Boolesch		
	veraltete_locales	Auflisten		
	Zugriffs-Gruppen-ID	Long		
	position	Ganzzahl		
	gefördert	Boolesch		
	Abschnitt-ID	Long		
	Quellgebietsschema	String		
	Name	String		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Titel	String		
	Benutzers egment_ID	Long		
	Content_Tags-ID	Auflisten		
	Anzahl der Stimmen	Ganzzahl		
	Stimmensumme	Ganzzahl		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	Name_Beze ichnung	String	EQUAL_TO	
Gruppe	URL	String		
	id	Long		
	is_public	Boolesch		
	Name	String		
	description	String		
	default	Boolesch		
	deleted	Boolesch		
	created_at	DateTime		
	updated_at	DateTime		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	ausschließen_gelöscht	Boolesch	EQUAL_TO	
-Automatisierung	URL	String		
	id	Long		
	Titel	String		
	aktiv	Boolesch		
	created_at	DateTime		
	updated_at	DateTime		
	default	Boolesch		
	actions	Auflisten		
	Positionen	Ganzzahl		
	Bedingungen	Struct		
	raw_title	String		
Gruppenmitgliedschaft	URL	String		
	id	Long		
	user_id	Long		
	group_id	Long		
	default	Boolesch		
	created_at	DateTime		
	updated_at	DateTime		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
Makro	URL	String		
	id	Long		
	Titel	String		
	aktiv	Boolesch	EQUAL_TO	
	created_at	DateTime		
	updated_at	DateTime		
	default	Boolesch		
	actions	Auflisten		
	position	Ganzzahl		
	description	String		
	raw_title	String		
	Einschränkung	Struct		
	access	String	EQUAL_TO	
	Kategorie	Ganzzahl	EQUAL_TO	
	group_id	Long	EQUAL_TO	
	nur_sichtbar	Boolesch	EQUAL_TO	
Organisationen	URL	String		
	id	Long		
	external_id	String		
	Name	String		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Domainnamen	Auflisten		
	Details	String		
	notes	String		
	group_id	Long		
	geteilte Tickets	Boolesch		
	geteilte Kommentare	Boolesch		
	tags	Auflisten		
	organisationsfelder	Struct		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	DML_STATUS	String		Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.
Zufriedenheitsbewertung	URL	String		
	id	Long		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	assignee_id	Long		
	Kommentar	String		
	group_id	Long		
	Grund	String		
	Ursachencode	Ganzzahl		
	Ursachen-ID	Long		
	requester_id	Long		
	Bewertung	String	EQUAL_TO	
	Ticket_ID	Ganzzahl		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	start_time	DateTime	EQUAL_TO	
	end_time	DateTime	EQUAL_TO	
	DML_STATUS	String		Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
Ticket-Aktivität	Akteur	Struct		
	Akteur_ID	Long		
	created_at	DateTime		
	id	Long		
	object	Struct		
	Ziel	Struct		
	Titel	String		
	updated_at	DateTime		
	URL	String		
	user	Struct		
	user_id	Long		
	verb	String		
	since	DateTime	EQUAL_TO	
Kommentar zum Ticket	id	Long		
	Typ	String		
	Autoren-ID	Long		
	body	String		
	html_body	String		
	schlichter_Körper	String		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	öffentlich	Boolesch		
	Anhänge	Auflisten		
	audit_id	Long		
	über	Struct		
	created_at	DateTime		
	Metadaten	Struct		
	ticket_id	Ganzzahl	EQUAL_TO	
	Inline_Bilder einschließen	Boolesch	EQUAL_TO	
Ticket-Veranstaltungen	id	Long		
	Ticket_ID	Long		
	Zeitstempel	Long		
	created_at	DateTime		
	Updater-ID	Long		
	Ereignisse für Kinder	Auflisten		
	über	String		
	system	Struct		
	event_type	String		
	comment_present	Boolesch		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	kommentar_öffentlich	Boolesch		
	via_reference_id	Long		
	created_at	DateTime	EQUAL_TO	
	DML_STATUS	String		Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.
Ticket-Feld	URL	String		
	id	Long		
	Typ	String		
	Titel	String		
	unformatierter Titel	String		
	description	String		
	rohe_beschreibung	String		
	position	Ganzzahl		
	aktiv	Boolesch		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Erforderlich	Boolesch		
	für Agenten zusammengebrochen	Boolesch		
	regexp_für_validierung	String		
	Titel_im_Portal	String		
	unformatierter Titel im Portal	String		
	sichtbar_im_Portal	Boolesch		
	im_Portal_bearbeitbar	Boolesch		
	im_Portal_erforderlich	Boolesch		
	Tag (Markierung)	String		
	created_at	DateTime		
	updated_at	DateTime		
	entfernbar	Boolesch		
	Beschreibung des Agenten	String		
	benutzerdefinierte_Feldoptionen	Auflisten		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	benutzerdefinierte_Status	Auflisten		
	Beziehungsfilter	Struct		
	Zieltyp der Beziehung	String		
	sub_type_id	Ganzzahl		
	system_field_options	Auflisten		
	locale	String	EQUAL_TO	
ticket-metric-events	id	Long		
	time	DateTime	EQUAL_TO	
	Ticket_ID	Ganzzahl		
	Metrik	String		
	instance_id	Ganzzahl		
	Typ	String		
	DML_STATUS	String	EQUAL_TO	Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
Ticket-Metrik	URL	String		
	id	Long		
	Ticket_ID	Ganzzahl		
	created_at	DateTime		
	updated_at	DateTime		
	group_stations	Ganzzahl		
	zugewiesene Stationen	Ganzzahl		
	wieder geöffnet	Ganzzahl		
	Antworten	Ganzzahl		
	assignee_ updated_at	DateTime		
	Anfordere r_aktualisiert_at	DateTime		
	anfänglic h_zugewie sen_am	DateTime		
	zugewiesen_am	DateTime		
	gelöst_am	DateTime		
	Letzter_K ommentar_ hinzugefügt_am	DateTime		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Antwortzeit_in_Minuten	Struct		
	Zeit für die erste Auflösung in Minuten	Struct		
	Zeit für die vollständige Auflösung in Minuten	Struct		
	Wartezeit für Agenten in Minuten	Struct		
	requester_wait_time_in_minutes	Struct		
	Wartezeit_in_Sekunden	Struct		
	Antwortzeit_in_Sekunden	Struct		
	custom_status_updated_at	DateTime		
Ticket überspringen	created_at	DateTime		
	id	Long		
	Grund	String		
	Fahrkarte	Struct		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Ticket_ID	Ganzzahl		
	updated_at	DateTime		
	user_id	Long		
Eintrittskarten	URL	String		
	id	Long		
	external_id	String	EQUAL_TO	
	Typ	String		
	subject	String		
	raw_subject	String		
	description	String		
	priority	String		
	Status	String		
	Empfänger	String		
	Anforderer	Struct		
	requester_id	Long		
	submitter_id	Long		
	assignee_id	Long		
	organization_id	Long		
	group_id	Long		
	collaborator_ids	Auflisten		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	E-Mails_cc_ids	Auflisten		
	follower_ids	Auflisten		
	forum_topic_id	Ling		
	problem_id	Long		
	has_incidents	Boolesch		
	due_at	DateTime		
	tags	Auflisten		
	über	Struct		
	custom_fields	Auflisten		
	satisfaction_rating	Struct		
	sharing_agreement_ids	Auflisten		
	followup_ids	Auflisten		
	via_followup_source_id	Long		
	ticket_form_id	Long		
	brand_id	Long		
	allow_channelback	Boolesch		
	allow_attachments	Boolesch		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	is_public	Boolesch		
	von_messaging_channel	Boolesch		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	E-Mail-Adresse des Beauftragten	String		
	attribute_value_ids	Auflisten		
	Mitarbeiter	Auflisten		
	Kommentar	Struct		
	custom_status_id	Long		
	email_ccs	Struct		
	Anhänger	Struct		
	macro_id	Long		
	makros_ids	Auflisten		
	Metadaten	Struct		
	sicheres_aktualisieren	Boolesch		
	aktualisierter_stempel	DateTime		
	via_id	Long		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	voice_comment	Struct		
	DML_STATUS	String		Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.
Trigger-Kategorie	URL	String		
	id	String		
	Name	String		
	updated_at	DateTime		
	created_at	DateTime		
	position	Ganzzahl		
trigger	URL	String		
	id	Long		
	Titel	String		
	aktiv	Boolesch	EQUAL_TO	
	updated_at	DateTime		
	created_at	DateTime		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	default	Boolesch		
	actions	Auflisten		
	Bedingungen	Struct		
	description	String		
	position	Ganzzahl		
	raw_title	String		
	Kategorie_ID	String	EQUAL_TO	
Benutzer	URL	String		
	id	Long		
	external_id	String	EQUAL_TO	
	email	String		
	aktiv	Boolesch		
	alias	String		
	chat_only	Boolesch		
	benutzerdefinierte_Rollen-ID	Long		
	Rollentyp	Ganzzahl		
	Details	String		
	last_login_at	DateTime		
	locale	String		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	locale_id	Ganzzahl		
	moderator	Boolesch		
	notes	String		
	Name	String		
	only_private_comments	Boolesch		
	organization_id	Long		
	default_group_id	Long		
	phone	String		
	photo	Struct		
	URL des entfernten Fotos	String		
	restricted_agent	Boolesch		
	role	String	EQUAL_TO	
	shared	Boolesch		
	gemeinsam genutzter Agent	Boolesch		
	Tag (Markierung)	Auflisten		
	signature	String		
	suspended	Boolesch		
	ticket_restriction	String		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	time_zone	String		
	iana_timezone			
	two_factor_auth_enabled			
	user_fields			
	verified	Boolesch		
	report_csv	Boolesch		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	Berechtigungssatz	Long	EQUAL_TO	
	shared_phone_number	Boolesch		
	DML_STATUS	String		Ein benutzerdefiniertes Feld, das verwendet wird, um den Status des Datensatzes zu verfolgen, der erstellt, aktualisiert und gelöscht wurde.
Ansicht	URL	String		
	id	Long		

Entität	Feld	Datentyp	Unterstützte Operatoren	Kommentare
	Titel	String		
	aktiv	Boolesch	EQUAL_TO	
	updated_at	DateTime		
	created_at	DateTime		
	default	Boolesch		
	position	Ganzzahl		
	description	String		
	Ausführung	Struct		
	Einschränkung	Struct		
	raw_title	String		
	Bedingungen	Struct		
	access	String	EQUAL_TO	
	group_id	Long	EQUAL_TO	

> **Note**
>
> Die Datentypen Struct und List werden in der Antwort des Konnektors in den Datentyp String konvertiert.

Abfragen partitionieren

Partitionen werden in Zendesk nicht unterstützt.

Zendesk-Verbindungsoptionen

Im Folgenden sind die Verbindungsoptionen für Zendesk aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Zendesk.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Zendesk Rest API-Version, die Sie verwenden möchten. Zum Beispiel: v2.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten. Zum Beispiel: id, name, url, created_at
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein. Zum Beispiel: group_id = 100
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage. Zum Beispiel: „SELECT id, url FROM users WHERE role=\" end-user\ "“
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll. Das Standardfeld ist `update_at` für Entitäten vorgesehen, die die inkrementelle Export-API unterstützen (`created_at`für `ticket-events` und `time` für`ticket-metric-events`).
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes. Optional; diese Option wird vom Konnektor verarbeitet, sofern sie nicht in der Joboption angegeben ist. Standardwert: „2024-05-01T 20:55:02.000 Z
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen. Optional; diese Option wird vom Connector verarbeitet, sofern sie nicht in der Job-Option angegeben ist. Standardwert: 1.
- `IMPORT_DELETED_RECORDS`(Zeichenfolge) — Standard: FALSE. Wird zum Lesen verwendet. Um die gelöschten Datensätze während der Abfrage abzurufen.
- `ACCESS_TOKEN`- Zugriffstoken, das in der Anfrage verwendet werden soll.
- `INSTANCE_URL`- URL der Instanz, in der der Benutzer die Operationen ausführen möchte. Zum Beispiel: https://{subdomain}/.zendesk.com

Einschränkungen

Im Folgenden sind die Einschränkungen des Zendesk-Connectors aufgeführt:

- Die Offset-basierte Paginierung begrenzt die Anzahl der Seiten, die abgerufen werden können, auf 100. Dies wird jedoch nicht empfohlen, da die Gesamtzahl der abgerufenen Datensätze 10.000 beträgt. Die cursorbasierte Paginierung, die für den Zendesk-Connector implementiert ist, überwindet diese Einschränkung jedoch. Nur der Filteroperator EQUAL_TO wird über die Zendesk-API unterstützt.

  Aufgrund dieser Einschränkung wird die Partitionierung für den Zendesk-Connector nicht unterstützt.
- Für die Entität „Ticket Event“ beträgt das Ratenlimit 10 Anfragen pro Minute. Beim Ausführen eines AWS Glue ETL-Jobs erhalten Sie möglicherweise die Fehlermeldung 429 (zu viele Anfragen).

Verbindung zu Zoho CRM herstellen

Zoho CRM fungiert als zentrales Repository, um Vertriebs-, Marketing- und Kundensupport-Aktivitäten zusammenzuführen und Prozesse, Richtlinien und Mitarbeiter auf einer Plattform zu optimieren. Zoho CRM kann einfach an die spezifischen Bedürfnisse jedes Unternehmenstyps und jeder Unternehmensgröße angepasst werden.

Die Entwicklerplattform von Zoho CRM bietet die richtige Mischung aus Low-Code- und Pro-Code-Tools für Unternehmen/Unternehmen, um die Arbeit zu automatisieren, Daten im gesamten Unternehmensstapel zu integrieren und maßgeschneiderte Lösungen für Web und Mobilgeräte zu erstellen.

Themen

- AWS Glue Unterstützung für Zoho CRM
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Zoho CRM
- Konfiguration von Zoho CRM-Verbindungen
- Aus Zoho CRM-Entitäten lesen
- Verbindungsoptionen für Zoho CRM
- Einschränkungen und Hinweise für den Zoho CRM-Konnektor

AWS Glue Unterstützung für Zoho CRM

AWS Glue unterstützt Zoho CRM wie folgt:

Wird es als Quelle unterstützt?

Ja — Sync und Async. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Zoho CRM abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen von Zoho CRM

Die folgenden Zoho CRM-API-Versionen werden unterstützt:

- v7

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Zoho CRM

Bevor Sie Daten aus Zoho CRM übertragen können, müssen Sie die folgenden Anforderungen erfüllen: AWS Glue

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Zoho CRM-Konto.
- Ihr Zoho CRM-Konto ist für den API-Zugriff aktiviert.
- Sie haben einen registrierten API-Client in der API-Konsole, um Anmeldeinformationen zu erhalten OAuth .

Konfiguration von Zoho CRM-Verbindungen

Die Art der Gewährung bestimmt, wie mit Zoho CRM AWS Glue kommuniziert wird, um Zugriff auf Ihre Daten anzufordern. Ihre Wahl wirkt sich auf die Anforderungen aus, die Sie erfüllen müssen, bevor Sie die Verbindung herstellen. Zoho CRM unterstützt nur den Grant-Typ AUTHORIZATION_CODE für 2.0. OAuth

- Dieser Gewährungstyp wird als „dreibeiniges“ angesehen, OAuth da er darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Die AWS Glue Konsole leitet den Benutzer zu Zoho CRM weiter, wo er sich anmelden

und Glue die angeforderten Berechtigungen für den Zugriff auf seine Zoho CRM-Instanz gewähren muss.

- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Zoho CRM zu erstellen und ihre eigene Client-ID, Authentifizierungs-URL, Token-URL und Instanz-URL anzugeben, wenn sie Verbindungen über die Konsole herstellen. AWS Glue In diesem Szenario werden sie weiterhin zu Zoho CRM weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen zu autorisieren AWS Glue .
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken bleibt eine Stunde lang gültig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Die öffentliche Dokumentation von Zoho CRM zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow finden Sie unter Authentifizierung.

So konfigurieren Sie eine Zoho CRM-Verbindung:

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Zoho CRM aus.
   b. Geben Sie die `INSTANCE_URL` Zoho CRM-Instanz an, zu der Sie eine Verbindung herstellen möchten.
   c. Geben Sie die Client-ID der Benutzeranwendung an.
   d. Wählen Sie die entsprechende Auth-URL aus der Dropdownliste aus.
   e. Wählen Sie die entsprechende Token-URL aus der Dropdownliste aus.
   f. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
```

```
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

g. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.

h. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.

2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.
3. Stellen `connectionName` Sie in Ihrer AWS Glue Jobkonfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus Zoho CRM-Entitäten lesen

Voraussetzung

Zoho CRM-Objekte, aus denen Sie lesen möchten. Sie benötigen den Objektnamen.

Unterstützte Entitäten für Sync-Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Produkt	Ja	Ja	Ja	Ja	Ja
Zitat	Ja	Ja	Ja	Ja	Ja
Bestellungen	Ja	Ja	Ja	Ja	Ja
Lösung	Ja	Ja	Ja	Ja	Ja
Rufen Sie	Ja	Ja	Ja	Ja	Ja
Aufgabe	Ja	Ja	Ja	Ja	Ja
Ereignis	Ja	Ja	Ja	Ja	Ja

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Rechnung	Ja	Ja	Ja	Ja	Ja
Account	Ja	Ja	Ja	Ja	Ja
Kontakt	Ja	Ja	Ja	Ja	Ja
Hersteller	Ja	Ja	Ja	Ja	Ja
Kampagne	Ja	Ja	Ja	Ja	Ja
Deal	Ja	Ja	Ja	Ja	Ja
Blei	Ja	Ja	Ja	Ja	Ja
Benutzerdefiniertes Modul	Ja	Ja	Ja	Ja	Ja
Kundenauftrag	Ja	Ja	Ja	Ja	Ja
Preisbücher	Ja	Ja	Ja	Ja	Ja
Case	Ja	Ja	Ja	Ja	Ja

Beispiel:

```
zoho_read = glueContext.create_dynamic_frame.from_options(
    connection_type="ZOHO",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "INSTANCE_URL": "https://www.zohoapis.in/"
    }
```

Unterstützte Entitäten für die Async-Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Produkt	Ja	Nein	Nein	Ja	Nein
Zitat	Ja	Nein	Nein	Ja	Nein
Bestellungen	Ja	Nein	Nein	Ja	Nein
Lösung	Ja	Nein	Nein	Ja	Nein
Rufen Sie	Ja	Nein	Nein	Ja	Nein
Aufgabe	Ja	Nein	Nein	Ja	Nein
Ereignis	Ja	Nein	Nein	Ja	Nein
Rechnung	Ja	Nein	Nein	Ja	Nein
Account	Ja	Nein	Nein	Ja	Nein
Kontakt	Ja	Nein	Nein	Ja	Nein
Hersteller	Ja	Nein	Nein	Ja	Nein
Kampagne	Ja	Nein	Nein	Ja	Nein
Deal	Ja	Nein	Nein	Ja	Nein
Blei	Ja	Nein	Nein	Ja	Nein
Benutzerdefiniertes Modul	Ja	Nein	Nein	Ja	Nein
Kundenauftrag	Ja	Nein	Nein	Ja	Nein

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Preisbücher	Ja	Nein	Nein	Ja	Nein
Case	Ja	Nein	Nein	Ja	Nein

Beispiel:

```
zoho_read = glueContext.create_dynamic_frame.from_options(
    connection_type="ZOHO",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "INSTANCE_URL": "https://www.zohoapis.in/",
        "TRANSFER_MODE": "ASYNC"
    }
```

Angaben zum Zoho CRM-Feld:

Zoho CRM bietet Endpunkte zum dynamischen Abrufen von Metadaten für unterstützte Entitäten. Daher wird die Bedienerunterstützung auf der Datentypebene erfasst.

Entität	Datentyp	Unterstützte Operatoren
Zoho-Entitäten (alle Entitäten)	Ganzzahl	! =, =, <, <=, >, >=, ZWISCHEN
	String	Wie, =,! =
	BigInteger	! =, =, <, <=, >, >=, ZWISCHEN
	Boolesch	=
	Double	! =, =, <, <=, >, >=, ZWISCHEN

Entität	Datentyp	Unterstützte Operatoren
	BigDecimal	! =, =, <, <=, >, >=, ZWISCHEN
	Datum	! =, =, <, <=, >, >=, ZWISCHEN
	DateTime	! =, =, <, <=, >, >=, ZWISCHEN
	Struct	N/A
	Auflisten	N/A

Abfragen partitionieren

Die Partitionierung wird im asynchronen Modus nicht unterstützt.

Filterbasierte Partitionierung (Sync-Modus):

Sie können die zusätzlichen Spark-OptionenPARTITION_FIELD,, und angeben LOWER_BOUNDUPPER_BOUND, NUM_PARTITIONS ob Sie Parallelität in Spark nutzen möchten. Mit diesen Parametern würde die ursprüngliche Abfrage in eine NUM_PARTITIONS Reihe von Unterabfragen aufgeteilt, die von Spark-Aufgaben gleichzeitig ausgeführt werden können.

- PARTITION_FIELD: der Name des Feldes, das zur Partitionierung der Abfrage verwendet werden soll.
- LOWER_BOUND: ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.

  Für das Datetime-Feld akzeptieren wir das Spark-Zeitstempelformat, das in Spark-SQL-Abfragen verwendet wird.

  Beispiele für gültige Werte:

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- UPPER_BOUND: ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- NUM_PARTITIONS: die Anzahl der Partitionen.

Beispiel:

```
zoho_read = glueContext.create_dynamic_frame.from_options(
    connection_type="zohocrm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "PARTITION_FIELD": "Created_Time"
        "LOWER_BOUND": "2022-01-01T01:01:01.000Z"
        "UPPER_BOUND": "2024-01-01T01:01:01.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Verbindungsoptionen für Zoho CRM

Im Folgenden sind die Verbindungsoptionen für Zoho CRM aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name Ihres Objekts in Zoho CRM.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Die Zoho CRM-Rest-API-Version, die Sie verwenden möchten.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Spalten, die Sie für das Objekt auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.
- `PARTITION_FIELD`(Zeichenfolge) — Wird zum Lesen verwendet. Feld, das zur Partitionierung der Abfrage verwendet werden soll.
- `LOWER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein inklusiver Untergrenzwert des ausgewählten Partitionsfeldes.
- `UPPER_BOUND`(Zeichenfolge) — Wird zum Lesen verwendet. Ein exklusiver Obergrenzwert des ausgewählten Partitionsfeldes.
- `NUM_PARTITIONS`(Ganzzahl) — Standard: 1. Wird zum Lesen verwendet. Anzahl der Partitionen zum Lesen.

- `INSTANCE_URL`(String) — (Erforderlich) Wird zum Lesen verwendet. Eine gültige URL für die Zoho CRM-Instanz.
- `TRANSFER_MODE`(Zeichenfolge) — Wird verwendet, um anzugeben, ob die Abfrage im asynchronen Modus ausgeführt werden soll.

Einschränkungen und Hinweise für den Zoho CRM-Konnektor

Im Folgenden finden Sie Einschränkungen oder Hinweise für den Zoho CRM-Connector:

- Mit API-Version v7 können Sie maximal 100.000 Datensätze abrufen. Weitere Informationen finden Sie in der Zoho-Dokumentation.
- Für die Event-Entität wird die Bezeichnung „Meeting“ angezeigt, wie in der Zoho-Dokumentation erwähnt.
- Für die Funktion „Alle auswählen“:
  - Sie können maximal 50 Felder von SaaS sowohl für den GET- als auch für den POST-Aufruf abrufen.
  - Wenn Sie Daten für ein bestimmtes Feld haben möchten, das nicht zu den ersten 50 Feldern gehört, müssen Sie die Liste der ausgewählten Felder manuell bereitstellen.
  - Wenn mehr als 50 Felder ausgewählt sind, werden alle Felder, die über die 50 Felder hinausgehen, gekürzt und enthalten in Amazon S3 Nulldaten.
  - Wenn im Fall eines Filterausdrucks die vom Benutzer bereitgestellte Liste mit 50 Feldern „id“ und „Created_Time“ nicht enthält, wird eine benutzerdefinierte Ausnahme ausgelöst, die den Benutzer auffordert, diese Felder einzubeziehen.
- Filteroperatoren können variieren, field-to-field obwohl sie denselben Datentyp haben. Daher müssen Sie für jedes Feld, das einen Fehler auf der SaaS-Plattform auslöst, manuell einen anderen Operator angeben.
- Für die Funktion „Sortieren nach“:
  - Daten können nur nach einem einzigen Feld ohne Filterausdruck sortiert werden, wohingegen Daten nach mehreren Feldern sortiert werden können, wenn ein Filterausdruck angewendet wird.
  - Wenn für das ausgewählte Feld keine Sortierreihenfolge angegeben ist, werden die Daten standardmäßig in aufsteigender Reihenfolge abgerufen.
- Die unterstützten Regionen für den Zoho CRM-Connector sind USA, Europa, Indien, Australien und Japan.
- Einschränkungen der asynchronen Lesefunktion:

- Limit-Order by und Partitionierung werden im asynchronen Modus nicht unterstützt.
- Im asynchronen Modus können wir Daten bis zu 500 Seiten mit 200.000 Datensätzen pro Seite übertragen.
- Für ein Intervall von einer Minute sind nur 10 Anfragen zum Herunterladen zulässig. Wenn Sie das Download-Limit überschreiten, gibt das System einen HTTP 429-Fehler zurück und pausiert alle Download-Anfragen für eine Minute, bevor die Verarbeitung fortgesetzt werden kann.
- Nach Abschluss des Sammelauftrags können Sie nur für einen Zeitraum von einem Tag auf die herunterladbare Datei zugreifen. Danach können Sie nicht mehr über Endpunkte auf die Datei zugreifen.
- Über einen Endpunkt können maximal 200 Auswahlfelder angegeben werden. Wenn Sie mehr als 200 Auswahlfelder in einem Endpunkt angeben, exportiert das System automatisch alle verfügbaren Felder für dieses Modul.
- Externe Felder, die in einem Modul erstellt wurden, werden beim Massenlesen nicht unterstützt APIs.
- Sortierung und `Group_by` Klauseln werden über diesen API-Endpunkt nicht unterstützt.
- Die Werte der Felder mit sensiblen Gesundheitsdaten werden nur abgerufen, wenn die Option Datenzugriff über API einschränken in den Compliance-Einstellungen deaktiviert ist. Wenn die Option aktiviert ist, ist der Wert im Ergebnis leer.
- Grenzwerte für Filterung und Kriterien
  - Die maximale Anzahl von Kriterien, die in einer Abfrage verwendet werden können, ist 25.
  - Filterung/Kriterien für mehrzeilige Textfelder werden nicht unterstützt.

Verbindung zu Zoom Meetings herstellen

Zoom Meetings ist eine Cloud-basierte Videokonferenzplattform, die für Videokonferenzen, Audiokonferenzen, Webinare, Besprechungsaufzeichnungen und Live-Chat verwendet werden kann.

Themen

- AWS Glue Unterstützung für Zoom Meetings
- Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten
- Konfiguration von Zoom-Meetings
- Konfiguration der Zoom Meetings-Client-App
- Konfiguration von Zoom Meetings-Verbindungen

- Aus Zoom Meetings-Entitäten lesen
- Verbindungsoptionen für Zoom Meetings
- Einschränkungen von Zoom Meetings

AWS Glue Unterstützung für Zoom Meetings

AWS Glue unterstützt Zoom Meetings wie folgt:

Wird es als Quelle unterstützt?

Ja. Sie können AWS Glue ETL-Jobs verwenden, um Daten aus Zoom Meetings abzufragen.

Wird es als Ziel unterstützt?

Nein.

Unterstützte API-Versionen von Zoom Meetings

Die folgenden Zoom Meetings-API-Versionen werden unterstützt:

- v2

Richtlinien, die die API-Operationen zum Erstellen und Verwenden von Verbindungen enthalten

Die folgende Beispielrichtlinie beschreibt die erforderlichen AWS IAM-Berechtigungen für das Erstellen und Verwenden von Verbindungen. Wenn Sie eine neue Rolle erstellen, erstellen Sie eine Richtlinie, die Folgendes enthält:

JSON

```
{
 "Version": "2012-10-17",
    "Statement": [
     {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
```

```
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Wenn Sie die obige Methode nicht verwenden möchten, verwenden Sie alternativ die folgenden verwalteten IAM-Richtlinien:

- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die verschiedene AWS Glue Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die in dieser Richtlinie angegebene Benennungskonvention für Ressourcen einhalten, verfügen AWS Glue Prozesse über die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Managementkonsole verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen.

Konfiguration von Zoom-Meetings

Bevor Sie die Übertragung von Daten aus Zoom Meetings verwenden AWS Glue können, müssen Sie die folgenden Anforderungen erfüllen:

Mindestanforderungen

Im Folgenden sind die Mindestanforderungen aufgeführt:

- Sie haben ein Zoom Meetings-Konto.
- Ihr Zoom-Konto ist für den API-Zugriff aktiviert.
- Sie haben in Ihrem Zoom Meetings-Konto eine OAuth2 App erstellt. Diese Integration stellt die Anmeldeinformationen bereit, die für den sicheren Zugriff auf Ihre Daten AWS Glue verwendet

werden, wenn authentifizierte Anrufe an Ihr Konto getätigt werden. Weitere Informationen finden Sie unter the section called “Konfiguration der Zoom Meetings-Client-App”.

Wenn Sie diese Anforderungen erfüllen, können Sie eine Verbindung AWS Glue zu Ihrem Zoom Meetings-Konto herstellen. Für typische Verbindungen müssen Sie in Zoom Meetings nichts weiter tun.

Konfiguration der Zoom Meetings-Client-App

1. Loggen Sie sich in den Zoom App Marketplace ein.
2. Wählen Sie Entwickeln > App erstellen.
3. Wählen Sie Allgemeine App für eine OAuth 2.0-basierte App.
4. Fügen Sie auf der Seite mit den Basisinformationen Informationen zur App hinzu oder aktualisieren Sie sie, z. B. den Namen der App, die Art und Weise, wie die App verwaltet wird, die App-Anmeldeinformationen und OAuth Informationen.
5. Bestätigen Sie im Abschnitt Wählen Sie, wie die App verwaltet wird, wie Ihre App verwaltet werden soll:
   a. Vom Administrator verwaltet: Kontoadministratoren fügen die App hinzu und verwalten sie
   b. Benutzerverwaltet: Einzelne Benutzer fügen die App hinzu und verwalten sie. Die App hat nur Zugriff auf die autorisierten Daten des Benutzers.
6. App-Anmeldeinformationen: Der Build-Flow generiert automatisch App-Anmeldeinformationen (Client-ID und Client-Geheimnis) für Ihre App.
7. Richten Sie im OAuth Informationsbereich die Einstellungen OAuth für Ihre App ein.
   a. OAuth Weiterleitungs-URL (erforderlich): Geben Sie Ihre Weiterleitungs-URL oder Ihren Endpunkt ein, um die OAuth Verbindung zwischen Ihrer App und Zoom einzurichten.
   b. Verwenden Sie die URL im strikten Modus (optional)
   c. Subdomain-Prüfung (optional)
   d. OAuth Zulassungslisten (erforderlich): Fügen Sie alle eindeutigen Einträge hinzu URLs , die Zoom als gültige Weiterleitungen für Ihre OAuth Flows zulassen soll.
8. Wählen Sie auf der Seite Bereiche die Zoom-API-Methoden aus, die Ihre App aufrufen darf. Die Bereiche definieren, welche Informationen und Funktionen Ihrem Benutzer zur Verfügung stehen. Wählen Sie die folgenden detaillierten Bereiche aus:
   - user:read:list_users:admin
   - zoom_rooms:read:list_rooms:admin

- group:read:list_members:admin
- Gruppe: Lesen: Administrator: Admin
- group:lesen:list_groups:admin
- bericht:lesen:admin
- role:read:list_roles, role:read:list_roles:admin

Sobald die Bereiche hinzugefügt wurden, wählen Sie Weiter und die App ist einsatzbereit.

Weitere Informationen zum OAuth 2.0-Setup finden Sie unter Integrationen (OAuth Apps).

Konfiguration von Zoom Meetings-Verbindungen

Zoom Meetings unterstützt den Zuschusstyp AUTHORIZATION_CODE für. OAuth2 Die Art des Zuschusses bestimmt, wie mit Zoom Meetings AWS Glue kommuniziert wird, um Zugriff auf Ihre Daten anzufordern.

- Diese Art der Gewährung wird als „dreibeiniges“ angesehen, OAuth da sie darauf beruht, dass Benutzer zur Authentifizierung des Benutzers an einen Autorisierungsserver eines Drittanbieters weitergeleitet werden. Er wird verwendet, wenn Verbindungen über die Konsole hergestellt werden. AWS Glue Der Benutzer, der eine Verbindung herstellt, muss OAuth zugehörige Informationen wie Client-ID und Client Secret für seine Zoom Meetings-Client-Anwendung angeben. Die AWS Glue Konsole leitet den Benutzer zu Zoom weiter, wo er sich anmelden und den angeforderten Berechtigungen für AWS Glue den Zugriff auf seine Zoom Meetings-Instanz gewähren muss.
- Benutzer können sich weiterhin dafür entscheiden, ihre eigene verbundene App in Zoom Meetings zu erstellen und ihre eigene Client-ID und ihr eigenes Client-Geheimnis anzugeben, wenn sie Verbindungen über die AWS Glue Konsole herstellen. In diesem Szenario werden sie weiterhin zu Zoom Meetings weitergeleitet, um sich anzumelden und den Zugriff auf ihre Ressourcen AWS Glue zu autorisieren.
- Dieser Gewährungstyp führt zu einem Aktualisierungstoken und einem Zugriffstoken. Das Zugriffstoken ist kurzlebig und kann mithilfe des Aktualisierungstokens automatisch ohne Benutzerinteraktion aktualisiert werden.
- Die öffentliche Dokumentation zu Zoom Meetings zur Erstellung einer verbundenen App für den Authorization Code OAuth Flow finden Sie unter Using OAuth 2.0.

So konfigurieren Sie eine Zoom Meetings-Verbindung:

1. Erstellen Sie in AWS Secrets Manager ein Secret mit den folgenden Details:
   a. Für die vom Kunden verwaltete verbundene App sollte das Secret das Consumer Secret der verbundenen App mit einem `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` Schlüssel enthalten.
   b. Hinweis: Sie müssen ein Geheimnis für Ihre Verbindungen in erstellen AWS Glue.

1. Erstellen Sie in AWS Glue Glue Studio eine Verbindung unter Datenverbindungen, indem Sie die folgenden Schritte ausführen:
   a. Wählen Sie bei der Auswahl eines Verbindungstyps Zoom Meetings aus.
   b. Geben Sie die Zoom Meetings-Umgebung an, zu der Sie eine Verbindung herstellen möchten.
   c. Wählen Sie die AWS IAM-Rolle aus, die die folgenden Aktionen übernehmen AWS Glue kann und über Berechtigungen verfügt:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

   d. Wählen Sie `secretName` die aus, in die Sie für diese Verbindung die Token einfügen AWS Glue möchten.
   e. Wählen Sie die Netzwerkoptionen aus, wenn Sie Ihr Netzwerk verwenden möchten.
2. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle `secretName` Leserechte.

Aus Zoom Meetings-Entitäten lesen

Voraussetzung

Ein Zoom Meetings-Objekt, aus dem Sie lesen möchten. Sie benötigen den Objektnamen wie `Group` oder`Zoom Rooms`.

Unterstützte Entitäten für die Quelle:

Entität	Kann gefiltert werden	Unterstützt Limit	Unterstützt Order by	Unterstützt Select *	Unterstützt Partitionierung
Zoom Rooms	Nein	Ja	Nein	Ja	Nein
Gruppe	Nein	Nein	Nein	Ja	Nein
Mitglied der Gruppe	Ja	Ja	Nein	Ja	Nein
Gruppenadministrator	Nein	Ja	Nein	Ja	Nein
Bericht (täglich)	Ja	Nein	Nein	Ja	Nein
Rollen	Nein	Nein	Nein	Ja	Nein
Benutzer	Ja	Ja	Nein	Ja	Nein

Beispiel:

```
zoom_read = glueContext.create_dynamic_frame.from_options(
    connection_type="zoom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "organization",
        "API_VERSION": "v2"
    }
)
```

Entitäts- und Felddetails von Zoom Meetings:

Zoom Meetings lädt dynamisch die verfügbaren Felder unter der ausgewählten Entität. Je nach Datentyp des Felds werden die folgenden Filteroperatoren unterstützt.

Entität	Feld	Datentyp	Unterstützte Operatoren	
Raum vergrößern	Status	String	=	
	Typ	String	=	
	unzugewiesene Räume	Boolesch	=	
	Standort-ID	String	=	
	Raum-ID	String	N/A	
	Aktivierungscode	String	N/A	
	id	String	N/A	
	Name	String	N/A	
	tag_ids	String	N/A	
	query_name	String	N/A	
Täglicher Bericht	Monat	Datum	=	
	date	Datum	N/A	
	Sitzungsprotokoll	Ganzzahl	N/A	
	Versammlungen	Ganzzahl	N/A	
	neue_Benutzer	Ganzzahl	N/A	
	Teilnehmer	Ganzzahl	N/A	
	group_id	String	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	
Benutzer	created_at	DateTime	N/A	
	Abt.	String	N/A	
	email	String	N/A	
	unique_id des Mitarbeiters	String	N/A	
	first_name	String	N/A	
	Gruppen-IDs	Auflisten	N/A	
	host_key	String	N/A	
	id	String	N/A	
	im_group_ids	String	N/A	
	letzte_Clientversion	String	N/A	
	last_login_time	DateTime	N/A	
	last_name	String	N/A	
	plan_united_type	String	N/A	
	benutzerdefinierte_Attribute	Auflisten	N/A	
	pmi	BigInteger	N/A	
	role_id	String	=	
	Status	String	=	

Entität	Feld	Datentyp	Unterstützte Operatoren	
	Zeitzone	String	N/A	
	Typ	Ganzzahl	N/A	
	verified	Ganzzahl	N/A	
	user_created_at	DateTime	N/A	
	Anzeige_name	String	N/A	
	phone_number	String	N/A	
	language	String	N/A	
	Lizenz	String	=	
Gruppe	id	String	N/A	
	Name	String	N/A	
	total_members	Ganzzahl	N/A	
Mitglied der Gruppe	email	String	N/A	
	first_name	String	N/A	
	id	String	N/A	
	last_name	String	N/A	
	Typ	Ganzzahl	N/A	
	primary_group	Boolesch	N/A	
	Mitglieds-ID	String	N/A	
Gruppenad ministrator	id	String	N/A	
	email	String	N/A	

Entität	Feld	Datentyp	Unterstützte Operatoren	
	Name	String	N/A	
role	description	String	N/A	
	id	String	N/A	
	Name	String	N/A	
	Gesamtzahl der Mitglieder	Ganzzahl	N/A	
	Typ	String	=	

Abfragen partitionieren

Zoom Meetings unterstützt keine filterbasierte Partitionierung oder datensatzbasierte Partitionierung.

Verbindungsoptionen für Zoom Meetings

Im Folgenden sind die Verbindungsoptionen für Zoom Meetings aufgeführt:

- `ENTITY_NAME`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Der Name der Zoom Meetings-Entität. Beispiel, `group`.
- `API_VERSION`(Zeichenfolge) — (Erforderlich) Wird zum Lesen verwendet. Zoom Meetings Rest-API-Version, die Sie verwenden möchten. Der Wert wird sein`v2`, da Zoom Meetings derzeit nur Version v2 unterstützt.
- `SELECTED_FIELDS`(Liste<String>) — Standard: leer (SELECT *). Wird zum Lesen verwendet. Eine durch Kommas getrennte Liste von Spalten, die Sie für die ausgewählte Entität auswählen möchten.
- `FILTER_PREDICATE`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Es sollte im Spark-SQL-Format sein.
- `QUERY`(Zeichenfolge) — Standard: leer. Wird zum Lesen verwendet. Vollständige Spark-SQL-Abfrage.

Einschränkungen von Zoom Meetings

Im Folgenden finden Sie Einschränkungen oder Hinweise für Zoom Meetings:

- Zoom Meetings unterstützt Orderby nicht.
- Zoom Meetings unterstützt keine filterbasierte Partitionierung, da es kein Feld gibt, das die erforderlichen Kriterien erfüllen kann.
- Zoom Meetings unterstützt keine datensatzbasierte Partitionierung, da das Paginierungslimit und die Offset-basierte Paginierung nicht unterstützt werden.

Hinzufügen einer JDBC-Verbindung mit Ihren eigenen JDBC-Treibern

Sie können Ihren eigenen JDBC-Treiber verwenden, wenn Sie eine JDBC-Verbindung verwenden. Wenn der vom AWS Glue Crawler verwendete Standardtreiber keine Verbindung zu einer Datenbank herstellen kann, können Sie Ihren eigenen JDBC-Treiber verwenden. Wenn Sie beispielsweise SHA-256 mit Ihrer Postgres-Datenbank verwenden möchten und ältere Postgres-Treiber dies nicht unterstützen, können Sie Ihren eigenen JDBC-Treiber verwenden.

Unterstützte Datenquellen

Unterstützte Datenquellen	Nicht unterstützte Datenquellen
MySQL	Snowflake
Postgres	
Oracle	
Redshift	
SQL Server	
Aurora*	

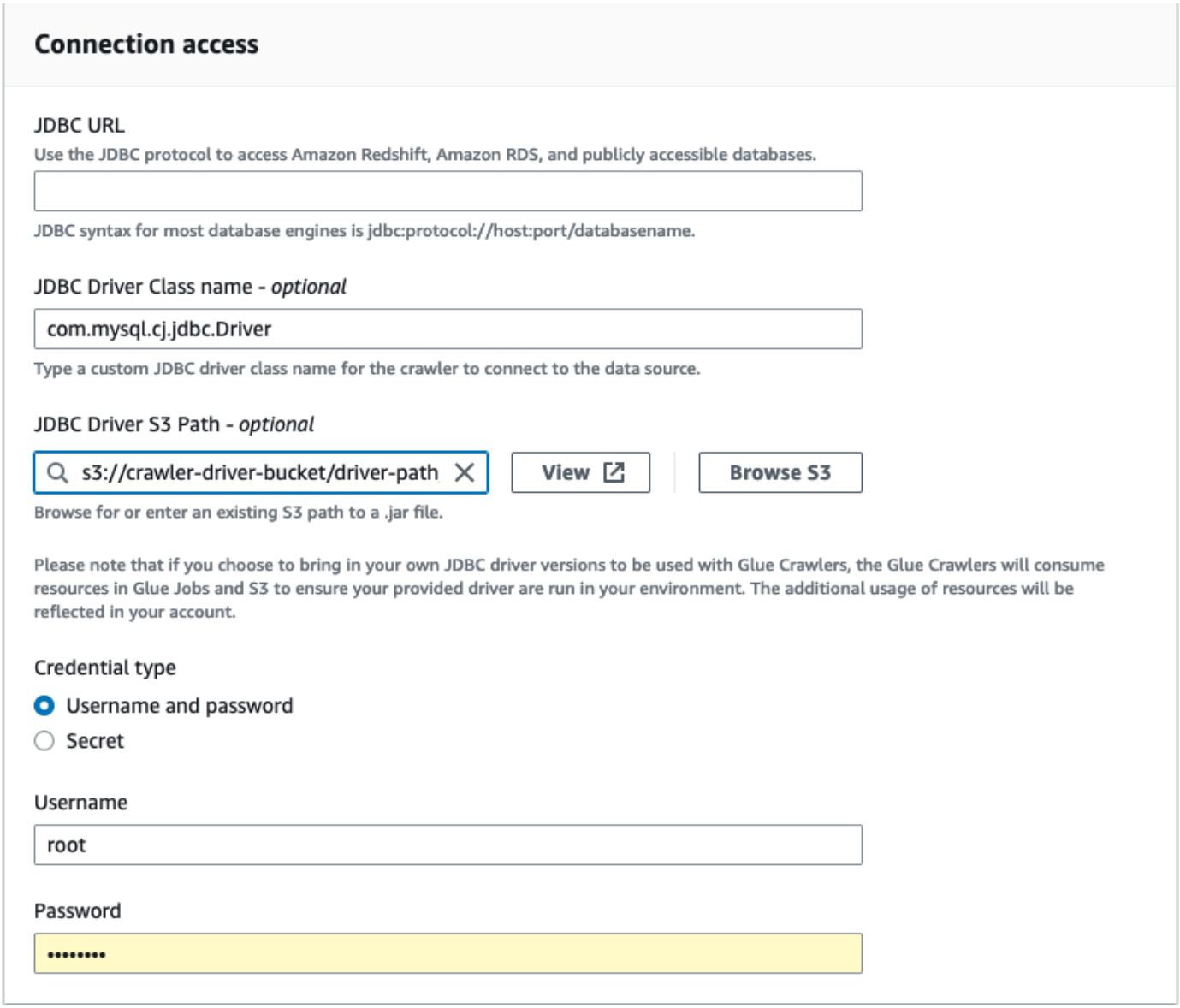
*Wird unterstützt, wenn der native JDBC-Treiber verwendet wird. Nicht alle Treiber-Features können genutzt werden.

Hinzufügen eines JDBC-Treibers zu einer JDBC-Verbindung

> **Note**
>
> Wenn Sie sich dafür entscheiden, Ihre eigenen JDBC-Treiberversionen zu verwenden, verbrauchen AWS Glue Crawler Ressourcen in AWS Glue Jobs und Amazon S3 S3-Buckets, um sicherzustellen, dass Ihr bereitgestellter Treiber in Ihrer Umgebung ausgeführt wird. Der zusätzliche Ressourcenverbrauch wird in Ihrem Konto angezeigt. Die Kosten für AWS Glue Crawler und Jobs fallen unter die Kategorie Abrechnung. AWS Glue Darüber hinaus bedeutet das Bereitstellen eines eigenen JDBC-Treibers nicht, dass der Crawler in der Lage ist, alle Features des Treibers zu nutzen.

So fügen Sie Ihren eigenen JDBC-Treiber zu einer JDBC-Verbindung hinzu:

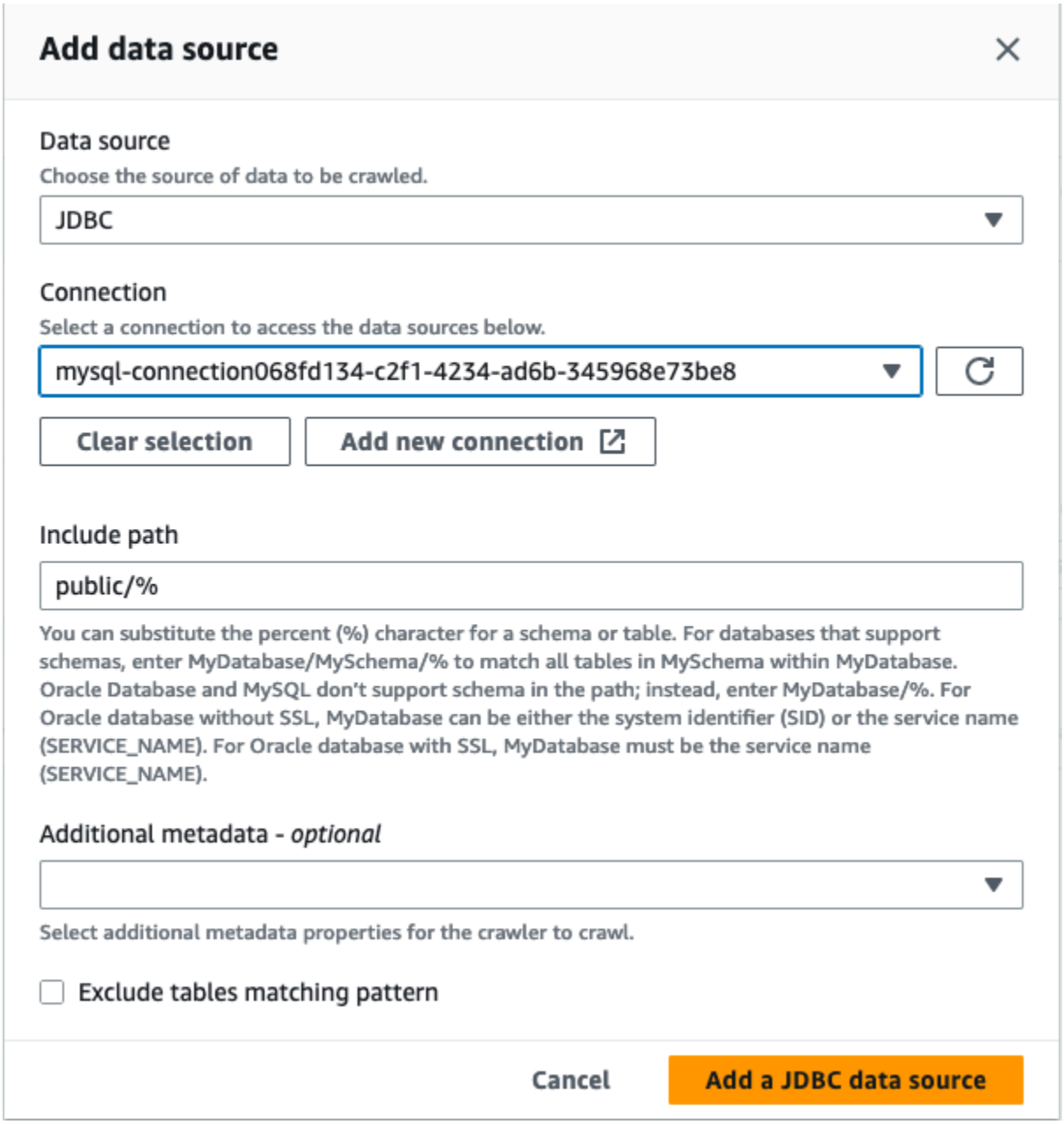
1. Fügen Sie die JDBC-Treiberdatei einem Amazon-S3-Speicherort hinzu. Sie können einen and/or Bucket-Ordner erstellen oder einen vorhandenen and/or Bucket-Ordner verwenden.
2. Wählen Sie in der AWS Glue Konsole im Menü auf der linken Seite unter Datenkatalog die Option Verbindungen aus und erstellen Sie dann eine neue Verbindung.
3. Füllen Sie die Felder für Verbindungseigenschaften aus und wählen Sie JDBC als Verbindungstyp aus.
4. Geben Sie unter Verbindungszugriff die JDBC-URL und den Namen der JDBC-Treiberklasse ein – optional. Der Name der Treiberklasse muss für eine Datenquelle stehen, die von Crawlern unterstützt wird. AWS Glue



5. Wählen Sie den Amazon-S3-Pfad, unter dem sich der JDBC-Treiber befindet, im Feld Amazon-S3-Pfad des JDBC-Treibers aus – optional.
6. Füllen Sie die Felder für den Anmeldeinformationstyp aus, wenn Sie einen Benutzernamen, ein Passwort oder ein Secret eingeben. Wenn der Vorgang abgeschlossen ist, wählen Sie Verbindung erstellen aus.

> **Note**
>
> Das Testen der Verbindung wird derzeit nicht unterstützt. Beim Crawling der Datenquelle mit einem von Ihnen bereitgestellten JDBC-Treiber überspringt der Crawler diesen Schritt.

7. Fügen Sie die neu erstellte Verbindung einem Crawler hinzu. Wählen Sie in der AWS Glue Konsole im linken Menü unter Datenkatalog die Option Crawler aus und erstellen Sie dann einen neuen Crawler.
8. Wählen Sie im Assistenten zum Hinzufügen von Crawlern in Schritt 2 die Option Datenquelle hinzufügen aus.



9. Wählen Sie JDBC als Datenquelle und wählen Sie die Verbindung aus, die in den vorherigen Schritten erstellt wurde. Complete
10. Um Ihren eigenen JDBC-Treiber mit einem AWS Glue Crawler zu verwenden, fügen Sie der vom Crawler verwendeten Rolle die folgenden Berechtigungen hinzu:

- Gewähren Sie Berechtigungen für die folgenden Auftragsaktionen: `CreateJob`, `DeleteJob`, `GetJob`, `GetJobRun`, `StartJobRun`.
- Gewähren Sie Berechtigungen für IAM-Aktionen: `iam:PassRole`
- Gewähren Sie Berechtigungen für alle Amazon-S3-Aktionen: `s3:DeleteObjects`, `s3:GetObject`, `s3:ListBucket`, `s3:PutObject`.
- Gewähren Sie dem Dienstprinzipal Zugriff auf bucket/folder in der IAM-Richtlinie.

Beispiel für eine IAM-Richtlinie:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/driver-parent-folder/
driver.jar",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Der AWS Glue Crawler erstellt zwei Ordner: _glue_job_crawler und _crawler.

Wenn sich die Treiber-JAR-Datei in dem Ordner befindet, fügen Sie die folgenden Ressourcen hinzu: `s3://amzn-s3-demo-bucket/driver.jar"`

```
"Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/_glue_job_crawler/*",
    "arn:aws:s3:::amzn-s3-demo-bucket/_crawler/*"
           ]
```

Wenn sich die Treiber-JAR-Datei in dem `s3://amzn-s3-demo-bucket/tmp/driver/subfolder/driver.jar"` Ordner befindet, fügen Sie die folgenden Ressourcen hinzu:

```
"Resource": [
              "arn:aws:s3:::amzn-s3-demo-bucket/tmp/_glue_job_crawler/*",
   "arn:aws:s3:::amzn-s3-demo-bucket/tmp/_crawler/*"
           ]
```

11. Wenn Sie eine VPC verwenden, müssen Sie den Zugriff auf den AWS Glue Endpunkt zulassen, indem Sie den Schnittstellenendpunkt erstellen und ihn Ihrer Routentabelle hinzufügen. Weitere Informationen finden Sie unter Erstellen eines Schnittstellen-VPC-Endpunkts für AWS Glue
12. Wenn Sie Verschlüsselung in Ihrem Datenkatalog verwenden, erstellen Sie den AWS KMS Schnittstellenendpunkt und fügen Sie ihn Ihrer Routentabelle hinzu. Weitere Informationen finden Sie unter Erstellen eines VPC-Endpunkts für AWS KMS.

Verwenden von benutzerdefinierten Konnektoren und Verbindungen mit AWS Glue Studio

AWS Glue bietet integrierte Unterstützung für die am häufigsten verwendeten Datenspeicher (wie Amazon Aurora Amazon Redshift, Microsoft SQL Server, MySQL, MongoDB und PostgreSQL) mithilfe von JDBC-Verbindungen. AWS Glue ermöglicht es Ihnen auch, benutzerdefinierte JDBC-Treiber in Ihren ETL-Jobs (Extrahieren, Transformieren und Laden) zu verwenden. Für Datenspeicher, die nicht nativ unterstützt werden, z. B. SaaS Anwendungen, können Sie Connectors verwenden.

Ein Connector ist ein optionales Codepaket, das den Zugriff auf Datenspeicher in unterstützt AWS Glue Studio. Sie können mehrere Konnektoren abonnieren, die in angeboten werden AWS Marketplace.

Beim Erstellen von ETL-Jobs können Sie einen nativ unterstützten Datenspeicher, einen Connector von AWS Marketplace oder Ihre eigenen benutzerdefinierten Konnektoren verwenden. Wenn Sie einen Connector verwenden, müssen Sie zunächst eine Verbindung für ihn erstellen. Eine Verbindung enthält die Eigenschaften, die für das Verbinden mit einem bestimmten Datenspeicher erforderlich sind. Sie verwenden die Verbindung mit Ihren Datenquellen und Datenzielen im ETL-Auftrag. Connectors und Verbindungen ermöglichen zusammen den Zugriff auf die Datenspeicher.

Die folgenden Verbindungen sind verfügbar, wenn Verbindungen für Konnektoren erstellt werden:

- Amazon Aurora – eine skalierbare, leistungsstarke relationale Datenbank-Engine mit integrierter Sicherheit, Sicherung und Wiederherstellung sowie speicherinterner Beschleunigung.
- Amazon DocumentDB — ein skalierbarer, hochverfügbarer und vollständig verwalteter Dokumentendatenbankservice, der MongoDB und SQL unterstützt. APIs
- Amazon Redshift— ein skalierbarer, hochverfügbarer und vollständig verwalteter Dokumentendatenbankdienst, der MongoDB und SQL APIs unterstützt.
- Azure SQL – ein cloudbasierter relationaler Datenbankservice von Microsoft Azure, der skalierbare, zuverlässige und sichere Datenspeicher- und Verwaltungsfunktionen bietet.
- Cosmos DB – ein weltweit verteilter Cloud-Datenbankservice von Microsoft Azure, der skalierbare, leistungsstarke Datenspeicher- und Abfragefunktionen bietet.
- Google BigQuery — ein serverloses Cloud-Data Warehouse zum Ausführen schneller SQL-Abfragen für große Datensätze.
- JDBC – ein relationales Datenbankmanagementsystem (RDBMS), das eine Java-API für die Verbindung und Interaktion mit Datenverbindungen verwendet.
- Kafka – eine Open-Source-Stream-Verarbeitungsplattform, die für Datenstreaming und Messaging in Echtzeit verwendet wird.
- MariaDB – ein von der Community entwickelter Fork von MySQL, der verbesserte Leistung, Skalierbarkeit und Funktionen bietet.
- MongoDB – eine plattformübergreifende dokumentenorientierte Datenbank, die eine hohe Skalierbarkeit, Flexibilität und Leistung bietet.
- MongoDB Atlas — ein cloudbasiertes Database-as-a-Service (DBaaS) -Angebot von MongoDB, das die Verwaltung und Skalierung von MongoDB-Bereitstellungen vereinfacht.
- Microsoft SQL Server – ein relationales Datenbankmanagementsystem (RDBMS) von Microsoft, das robuste Datenspeicher-, Analyse- und Berichtsfunktionen bietet.
- Mixpanel — eine Analyseplattform, mit der Unternehmen analysieren können, wie Benutzer mit ihren Websites, mobilen Anwendungen und anderen digitalen Produkten interagieren.

- MySQL – ein relationales Open-Source-Datenbankmanagementsystem (RDBMS), das häufig in Webanwendungen verwendet wird und für seine Zuverlässigkeit und Skalierbarkeit bekannt ist.
- Netzwerk – Eine Netzwerkdatenquelle stellt eine über das Netzwerk zugängliche Ressource oder einen Service dar, auf den über eine Datenintegrationsplattform zugegriffen werden kann.
- OpenSearch— Eine OpenSearch Datenquelle ist eine Anwendung, mit der eine Verbindung hergestellt und Daten von dieser aufgenommen werden OpenSearch können.
- Oracle – ein relationales Datenbankmanagementsystem (RDBMS) von Oracle, das robuste Datenspeicher-, Analyse- und Berichtsfunktionen bietet.
- PostgreSQL – ein relationales Open-Source-Datenbankmanagementsystem (RDBMS), das robuste Datenspeicher-, Analyse- und Berichtsfunktionen bietet.
- Salesforce — Salesforce bietet Software für das Kundenbeziehungsmanagement (CRM), die Sie bei Vertrieb, Kundenservice, E-Commerce und mehr unterstützt. Wenn Sie ein Salesforce-Benutzer sind, können Sie eine Verbindung AWS Glue zu Ihrem Salesforce-Konto herstellen. Anschließend können Sie Salesforce als Datenquelle oder Ziel in Ihren ETL-Jobs verwenden. Führen Sie diese Jobs aus, um Daten zwischen Salesforce und AWS Services oder anderen unterstützten Anwendungen zu übertragen.
- SAP HANA – eine In-Memory-Datenbank und Analyseplattform, die schnelle Datenverarbeitung, fortschrittliche Analysen und Datenintegration in Echtzeit ermöglicht.
- Snowflake – ein cloudbasiertes Data Warehouse, das skalierbare, leistungsstarke Datenspeicher- und Analyseservices bietet.
- Teradata – ein relationales Datenbankmanagementsystem (RDBMS), das leistungsstarke Datenspeicher-, Analyse- und Berichtsfunktionen bietet.
- Vertica – ein spaltenorientiertes analytisches Data Warehouse, das für Big-Data-Analytik entwickelt wurde und schnelle Abfrageleistung, erweiterte Analysen und Skalierbarkeit bietet.

Erstellen von benutzerdefinierten Connectors

Sie können auch Ihren eigenen Connector erstellen und dann den Connector-Code hochladen AWS Glue Studio.

Benutzerdefinierte Konnektoren sind integriert in AWS Glue Studio durch die AWS Glue Spark-Laufzeit-API. Das Tool AWS Glue Mit Spark Runtime können Sie jeden Anschluss anschließen, der mit der Spark-, Athena- oder JDBC-Schnittstelle kompatibel ist. So können Sie jede Verbindungsoption übergeben, die mit dem benutzerdefinierten Connector verfügbar ist.

Sie können alle Ihre Verbindungseigenschaften mit kapseln AWS Glue Verbindungen und geben Sie den Verbindungsnamen für Ihren ETL-Job an. Durch die Integration mit Data-Catalog-Verbindungen können Sie dieselben Verbindungseigenschaften über mehrere Anrufe in einer einzelnen Spark-Anwendung oder über verschiedene Anwendungen hinweg verwenden.

Sie können zusätzliche Optionen für die Verbindung angeben. Das Job-Skript, das AWS Glue Studio generiert einen `Datasource` Eintrag, der die Verbindung verwendet, um Ihren Connector mit den angegebenen Verbindungsoptionen anzuschließen. Zum Beispiel:

```
Datasource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"dbTable":"Account","connectionName":"my-custom-
jdbc-
connection"}, transformation_ctx = "DataSource0")
```

Um einen benutzerdefinierten Konnektor hinzuzufügen AWS Glue Studio

1. Erstellen Sie den Code für Ihren benutzerdefinierten Connector. Weitere Informationen finden Sie unter Entwickeln von benutzerdefinierten Connectors.
2. Unterstützung hinzufügen für AWS Glue Funktionen für Ihren Connector. Hier sind einige Beispiele für diese Funktionen und wie sie in dem Job-Skript verwendet werden, das von generiert wurde AWS Glue Studio:

   - Datentypmapping – Ihr Connector kann den Typ von Spalten umwandeln, während sie aus dem zugrunde liegenden Datenspeicher gelesen werden. Beispiel: Ein `dataTypeMapping` von `{"INTEGER":"STRING"}` konvertiert alle Spalten des Typs `Integer` in Spalten vom Typ `String`, während die Datensätze analysiert und der `DynamicFrame` konstruiert werden. Dies hilft Benutzern, Spalten in Typen ihrer Wahl umzuwandeln.

     ```
     DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
     = "custom.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"}",
     connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
     ```

   - Partitionierung für parallel Lesevorgänge — AWS Glue ermöglicht parallel Datenlesevorgänge aus dem Datenspeicher, indem die Daten in einer Spalte partitioniert werden. Sie müssen die Partitionsspalte, die untere Partitionsgrenze, die obere Partitionsgrenze und die Anzahl der Partitionen angeben. Mit diesem Feature können Sie Datenparallelität und mehrere Spark Executors verwenden, die der Spark-Anwendung zugewiesen sind.

     ```
     DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
     ```

```
= "custom.jdbc", connection_options = {"upperBound":"200","numPartitions":"4",
"partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-jdbc"},
transformation_ctx = "DataSource0")
```

- AWS Secrets Manager Zum Speichern von Anmeldeinformationen verwenden — Die Datenkatalogverbindung kann auch einen `secretId` Schlüssel enthalten, der in AWS Secrets Manager gespeichert ist. In dem AWS Secret können Authentifizierungs- und Anmeldeinformationen sicher gespeichert und zur AWS Glue Laufzeit bereitgestellt werden. Alternativ können Sie auch die `secretId` wie folgt aus dem Spark-Skript angeben:

```
DataSource = glueContext.create_dynamic_frame.from_options(connection_type
= "custom.jdbc", connection_options = {"connectionName":"test-connection-jdbc",
 "secretId"-> "my-secret-id"}, transformation_ctx = "DataSource0")
```

- Filtern der Quelldaten mit Zeilenprädikaten und Spaltenprojektionen — Das AWS Glue Die Spark-Laufzeit ermöglicht es Benutzern auch, SQL-Abfragen zu starten, um Daten an der Quelle mit Zeilenprädikaten und Spaltenprojektionen zu filtern. Dadurch kann Ihr ETL-Auftrag gefilterte Daten schneller aus Datenspeichern laden, die Push-Downs unterstützen. Eine Beispiel-SQL-Abfrage, die an eine JDBC-Datenquelle weitergegeben wird, lautet:`SELECT id, name, department FROM department WHERE id < 200.`

```
DataSource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"query":"SELECT id, name, department FROM
 department
WHERE id < 200","connectionName":"test-connection-jdbc"}, transformation_ctx =
"DataSource0")
```

- Job-Lesezeichen — AWS Glue unterstützt das inkrementelle Laden von Daten aus JDBC-Quellen. AWS Glue verfolgt den zuletzt verarbeiteten Datensatz aus dem Datenspeicher und verarbeitet neue Datensätze in den nachfolgenden ETL-Jobausführungen. Auftragslesezeichen verwenden den Primärschlüssel als Standardspalte für den Lesezeichenschlüssel, vorausgesetzt, diese Spalte wird sequenziell vergrößert oder verringert. Weitere Informationen zu Auftragslesezeichen finden Sie unter Auftragslesezeichen im AWS Glue -Entwicklerhandbuch.

```
DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"jobBookmarkKeys":["empno"],
 "jobBookmarkKeysSortOrder"
:"asc", "connectionName":"test-connection-jdbc"}, transformation_ctx =
 "DataSource0")
```

3. Verpacken Sie den benutzerdefinierten Connector als JAR-Datei und laden Sie die Datei in Amazon S3 hoch.
4. Testen Sie Ihren benutzerdefinierten Connector. Weitere Informationen finden Sie in den Anweisungen GitHub unter Glue Custom Connectors: Leitfaden für lokale Validierungstests.
5. Im AWS Glue Studio Konsole: Wählen Sie im Navigationsbereich der Konsole Connectors aus.
6. Wählen Sie auf der Seite Connectors die Option Create custom Connector (Benutzerdefinierten Connector erstellen) aus.
7. Geben Sie auf der Seite Create custom Connector (Benutzerdefinierten Connector erstellen) Folgendes ein:
   - Den Speicherpfad der JAR-Datei mit dem benutzerdefinierten Code in Amazon S3.
   - Ein Name für den Connector, der verwendet wird von AWS Glue Studio.
   - Den Connector-Typ, bei dem es sich um JDBC, Spark oder Athena handeln kann.
   - Der Name des Einstiegspunkts in Ihrem benutzerdefinierten Code, der AWS Glue Studio ruft auf, den Konnektor zu verwenden.
     - Für JDBC-Connectors sollte in diesem Feld der Klassenname Ihres JDBC-Treibers eingetragen sein.
     - Bei Spark-Connectors sollte es sich bei diesem Feld um den vollqualifizierten Namen der Datenquellenklasse oder ihren Alias handeln, den Sie beim Laden der Spark-Datenquelle mit dem Operator `format` verwenden.
   - (Nur JDBC) Die Basis-URL, die von der JDBC-Verbindung für den Datenspeicher verwendet wird.
   - (Optional) Eine Beschreibung des benutzerdefinierten Connectors.
8. Wählen Sie Create Connector (Connector erstellen) aus.
9. Erstellen Sie auf der Seite Connectors (Verbindungen) eine Verbindung, die diesen Connector verwendet, wie in Erstellen von Verbindungen für Connectors beschrieben.

Konnektoren werden hinzugefügt zu AWS Glue Studio

Ein Konnektor ist ein Code, der die Kommunikation zwischen Ihrem Datenspeicher und AWS Glue. Sie können entweder einen Connector abonnieren, der unter angeboten wird AWS Marketplace, oder Sie können Ihren eigenen benutzerdefinierten Connector erstellen.

Konnektoren abonnieren AWS Marketplace

AWS Glue Studio macht es einfach, Konnektoren von AWS Marketplace hinzuzufügen.

Um einen Konnektor von AWS Marketplace bis hinzuzufügen AWS Glue Studio

1. Im AWS Glue Studio Konsole: Wählen Sie im Navigationsbereich der Konsole Connectors aus.
2. Wählen Sie auf der Seite Connectors die Option Go to AWS Marketplace (Zu MKT wechseln) aus.
3. Wählen Sie unter Ausgewählte Produkte den Connector aus, den Sie verwenden möchten. AWS Marketplace Sie können einen der empfohlenen Konnektoren auswählen oder die Suche verwenden. Sie können nach dem Namen oder Typ des Connectors suchen und Optionen verwenden, um die Suchergebnisse zu verfeinern.

   Wenn Sie einen der ausgewählten Konnektoren verwenden möchten, wählen Sie Produkt anzeigen aus. Wenn Sie die Suche verwendet haben, um einen Connector zu finden, wählen Sie den Namen des Connectors aus.
4. Die Registerkarten auf der Produktseite für den Connector enthalten Informationen zum Connector. Wenn Sie sich entscheiden, diesen Connector zu kaufen, wählen Sie Continue to Subscribe (Weiter zum Abonnement) aus.
5. Geben Sie die Zahlungsinformationen ein und wählen Sie dann Continue to Configure (Weiter zur Konfiguration) aus.
6. Klicken Sie auf der Seite Configure this software (Diese Software konfigurieren) die Bereitstellungsmethode und die Version des Connectors aus, die verwendet werden soll. Wählen Sie dann Continue to Launch (Weiter zum Start) aus.
7. Auf der Seite Launch this software (Diese Software starten) können Sie die Usage Instructions (Nutzungsanweisungen) des Connector-Anbieters ansehen. Wenn Sie bereit sind, fortzufahren, wählen Sie Verbindung aktivieren in AWS Glue Studio.

   Nach kurzer Zeit zeigt die Konsole die Seite Marketplace-Verbindung erstellen in AWS Glue Studio.
8. Erstellen Sie eine Verbindung, die diesen Connector verwendet, wie in Erstellen von Verbindungen für Connectors beschrieben.

   Alternativ können Sie auch Activate connector only (Nur Connector aktivieren) auswählen, um das Erstellen einer Verbindung zu überspringen. Sie müssen eine Verbindung zu einem späteren Zeitpunkt erstellen, bevor Sie den Connector verwenden können.

Erstellen von Verbindungen für Connectors

Eine AWS Glue Verbindung ist ein Datenkatalogobjekt, das Verbindungsinformationen für einen bestimmten Datenspeicher speichert. In Verbindungen werden Anmeldeinformationen, URI-Zeichenfolgen, Virtual Private Cloud (VPC)-Informationen und vieles mehr gespeichert. Das Erstellen von Verbindungen im Data Catalog spart den Aufwand, alle Verbindungsdetails jedes Mal anzugeben, wenn Sie einen Auftrag erstellen.

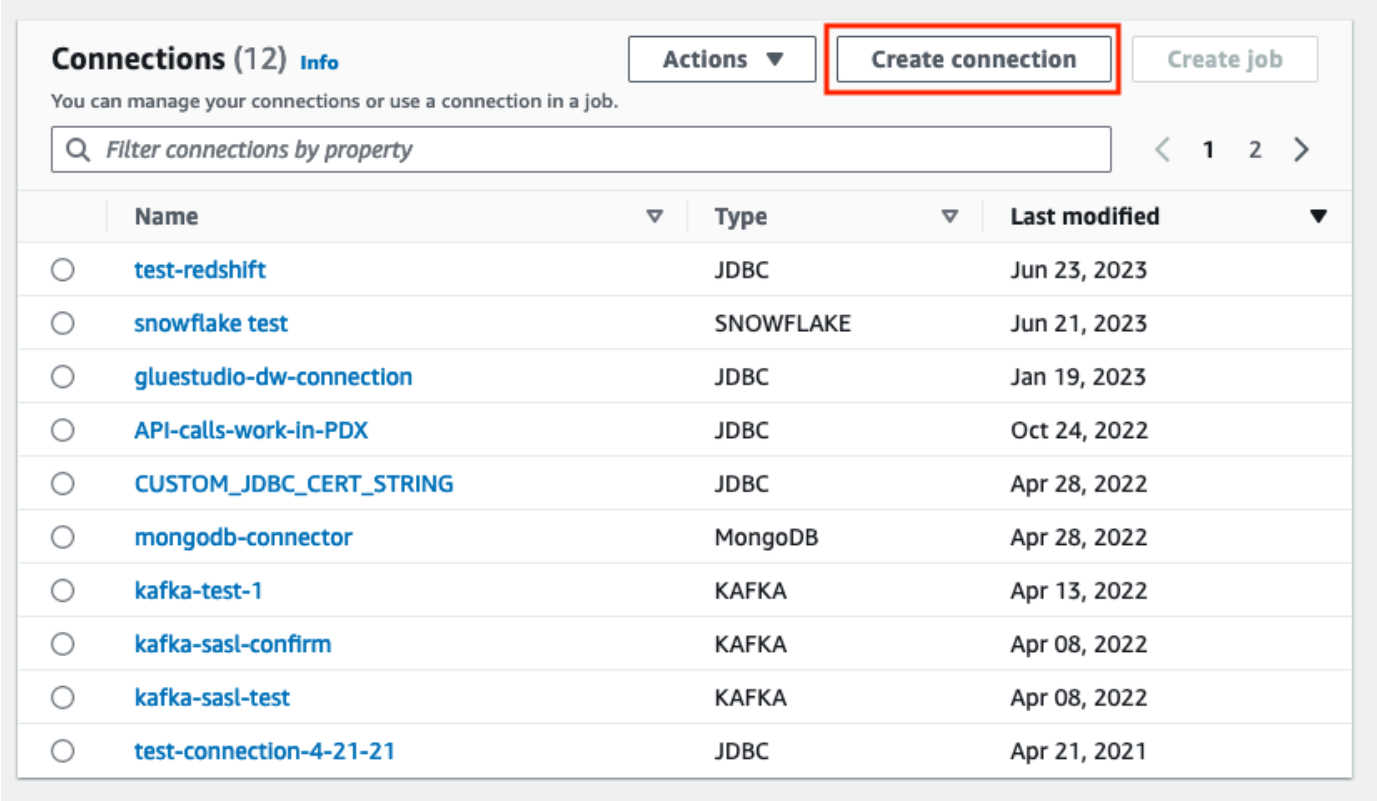
Eine Verbindung für einen Connector erstellen

1. Wählen Sie in der AWS Glue Studio Konsole im Navigationsbereich der Konsole Connectors aus. Wählen Sie im Abschnitt Verbindungen die Option Verbindung erstellen aus.
2. Wählen Sie in Schritt 1 des Datenverbindung erstellen-Assistenten die Datenquelle aus, für die Sie eine Verbindung erstellen möchten. Es gibt mehrere Möglichkeiten, die verfügbaren Datenquellen anzuzeigen. Unter anderem Folgende:

   - Filtern Sie die verfügbaren Datenquellen durch die Auswahl einer Registerkarte. Standardmäßig ist Alle Konnektoren ausgewählt.
   - Aktivieren Sie Liste, um die Datenquellen als Liste anzuzeigen, oder wechseln Sie zurück zu Raster, um die verfügbaren Connectors im Rasterlayout anzuzeigen.
   - Mit der Suchleiste können Sie die Liste der Datenquellen eingrenzen. Während der Eingabe werden Suchtreffer angezeigt und nicht übereinstimmende Quellen werden aus der Ansicht entfernt.

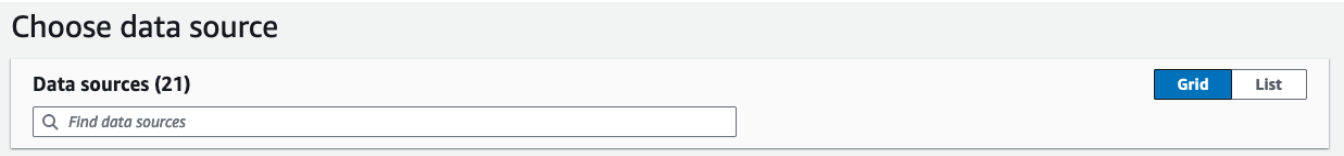
   Nachdem Sie die Datenquelle ausgewählt haben, gehen Sie auf Weiter.
3. Konfigurieren Sie die Verbindung in Schritt 2 des Assistenten.

   Geben Sie die Verbindungsdetails ein. Je nach ausgewähltem Verbindungstyp werden Sie aufgefordert, zusätzliche Informationen einzugeben:



4. Wählen Sie in Schritt 1 des Datenverbindung erstellen-Assistenten die Datenquelle aus, für die Sie eine Verbindung erstellen möchten. Es gibt mehrere Möglichkeiten, die verfügbaren Datenquellen anzuzeigen. Standardmäßig werden Ihnen alle verfügbaren Datenquellen in einem Rasterlayout angezeigt. Sie können auch:

   - Aktivieren Sie Liste, um die Datenquellen als Liste anzuzeigen, oder wechseln Sie zurück zu Raster, um die verfügbaren Connectors im Rasterlayout anzuzeigen.
   - Mit der Suchleiste können Sie die Liste der Datenquellen eingrenzen. Während der Eingabe werden Suchtreffer angezeigt und nicht übereinstimmende Quellen werden aus der Ansicht entfernt.

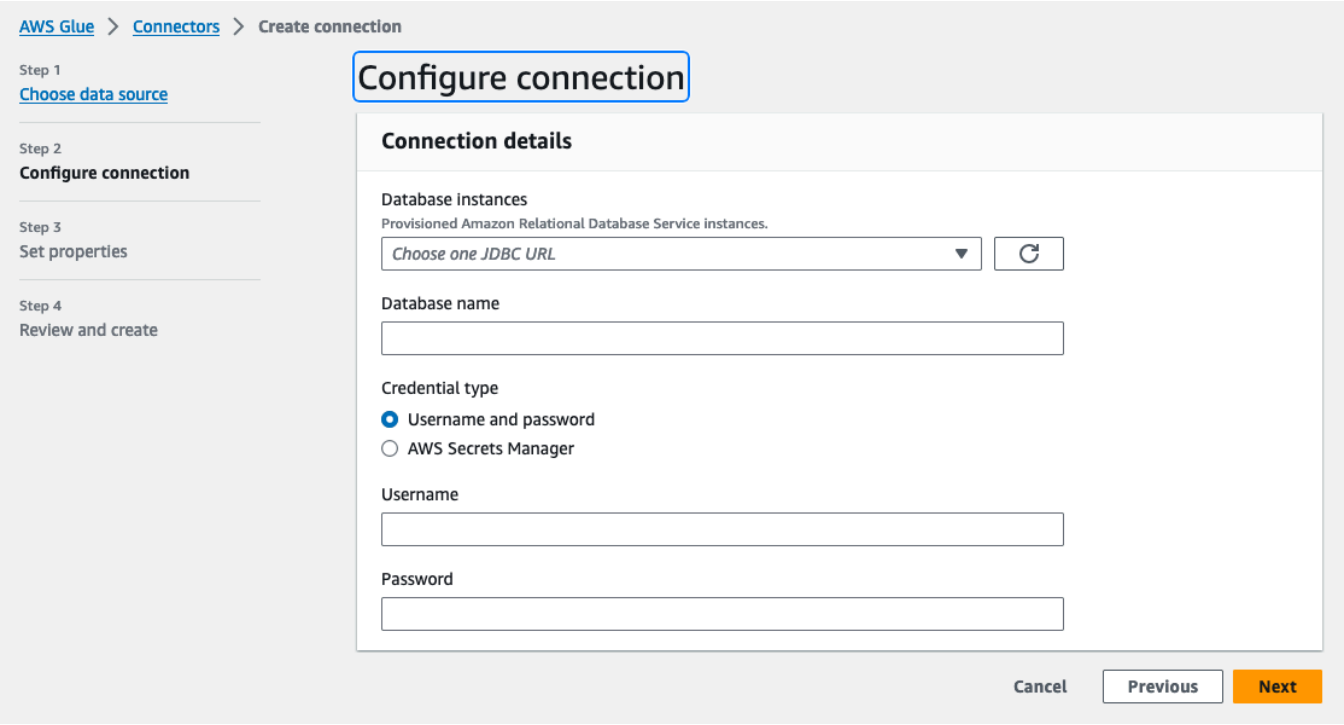


   Nachdem Sie die Datenquelle ausgewählt haben, gehen Sie auf Weiter.

5. Konfigurieren Sie die Verbindung in Schritt 2 des Assistenten.

Geben Sie die Verbindungsdetails ein. Je nachdem, welchen Verbindungstyp Sie ausgewählt haben, werden Sie möglicherweise aufgefordert, zusätzliche Informationen einzugeben: Dazu können folgende Angaben zählen:

- Verbindungsdetails – Diese Felder ändern sich je nach der Datenquelle, zu der Sie eine Verbindung herstellen. Wenn Sie beispielsweise eine Verbindung zu Amazon-DocumentDB-Datenbanken herstellen, geben Sie die Amazon-DocumentDB-URL ein. Wenn Sie eine Verbindung herstellen Amazon Aurora, wählen Sie die Datenbankinstanz aus und geben den Datenbanknamen ein. Die folgenden Verbindungsdetails sind erforderlich für Amazon Aurora:



- Anmeldeinformationstyp: Sie haben die Wahl zwischen Benutzername und Passwort oder AWS Secrets Manager. Geben Sie die angeforderten Authentifizierungsinformationen ein.
- Geben Sie für Connectors, die JDBC verwenden, die Informationen ein, die zum Erstellen der JDBC-URL für den Datenspeicher erforderlich sind.
- Wenn Sie eine Virtual Private Cloud (VPC) verwenden, geben Sie die Netzwerkinformationen für Ihre VPC ein.

6. Legen Sie in Schritt 3 des Assistenten die Verbindungseigenschaften fest. Optional können Sie bei diesem Schritt eine Beschreibung und Tags hinzufügen. Der Name ist erforderlich und ist bereits mit einem Standardwert gefüllt. Wählen Sie Weiter.

7. Überprüfen Sie die Verbindungsquelle, die Details und die Eigenschaften. Wenn Sie Änderungen vornehmen müssen, wählen Sie für den entsprechenden Schritt im Assistenten die Option Bearbeiten aus. Wenn Sie fertig sind, wählen Sie Verbindung erstellen aus.

   Wählen Sie Create Connection (Verbindung erstellen) aus.

   Sie werden zur Seite Connectors zurückgeleitet und das Informationsbanner gibt die Verbindung an, die erstellt wurde. Sie können die Verbindung jetzt in Ihren AWS Glue Studio -Aufträgen verwenden.

Erstellen einer Kafka-Verbindung

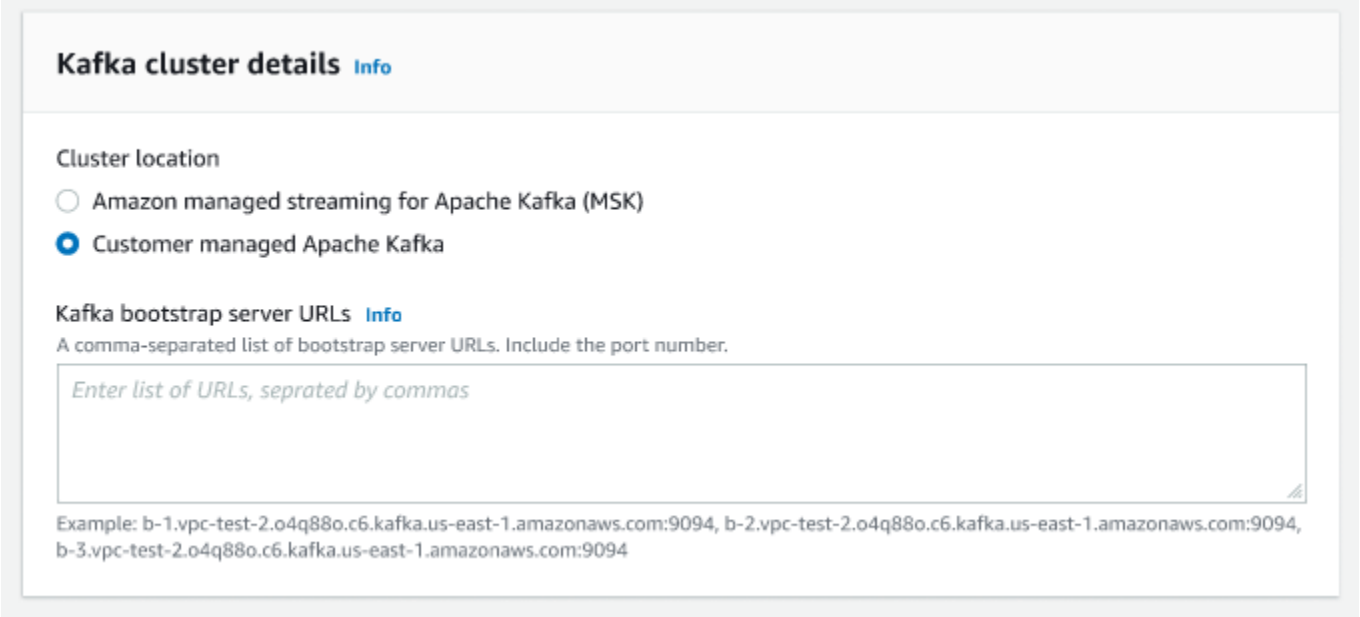
Wenn Sie eine Kafka-Verbindung erstellen, wählen Sie Kafka im Dropdown-Menü und es werden zusätzliche zu konfigurierende Einstellungen angezeigt:

- Details zum Kafka-Cluster
- Authentifizierung
- Verschlüsselung
- Netzwerkoptionen

Konfigurieren der Details des Kafka-Clusters

1. Wählen Sie den Clusterstandort aus. Sie können zwischen einem von Amazon verwalteten Streaming für Apache Kafka (MSK)-Cluster oder einem vom Kunden verwalteten Apache Kafka-Cluster wählen. Weitere Informationen zu von Amazon verwaltetes Streaming für Apache Kafka finden Sie unter Von Amazon verwaltetes Streaming für Apache Kafka (MSK).

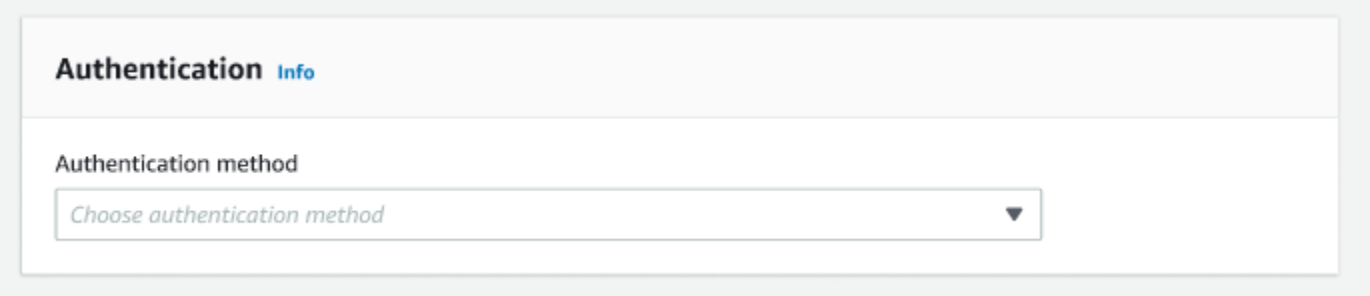
   > **Note**
   >
   > Amazon Managed Streaming for Apache Kafka unterstützt nur TLS- und SASL/SCRAM-SHA-512-Authentifizierungsmethoden.



2. Geben Sie den URLs für Ihre Kafka-Bootstrap-Server ein. Sie können mehrere eingeben, indem Sie jeden Server durch ein Komma trennen. Geben Sie die Portnummer am Ende der URL an, indem Sie `:<port number>` anfügen.

   Zum Beispiel: `b-1.vpc-test-2.034a88o.kafka-us-east-1.amazonaws.com:9094`

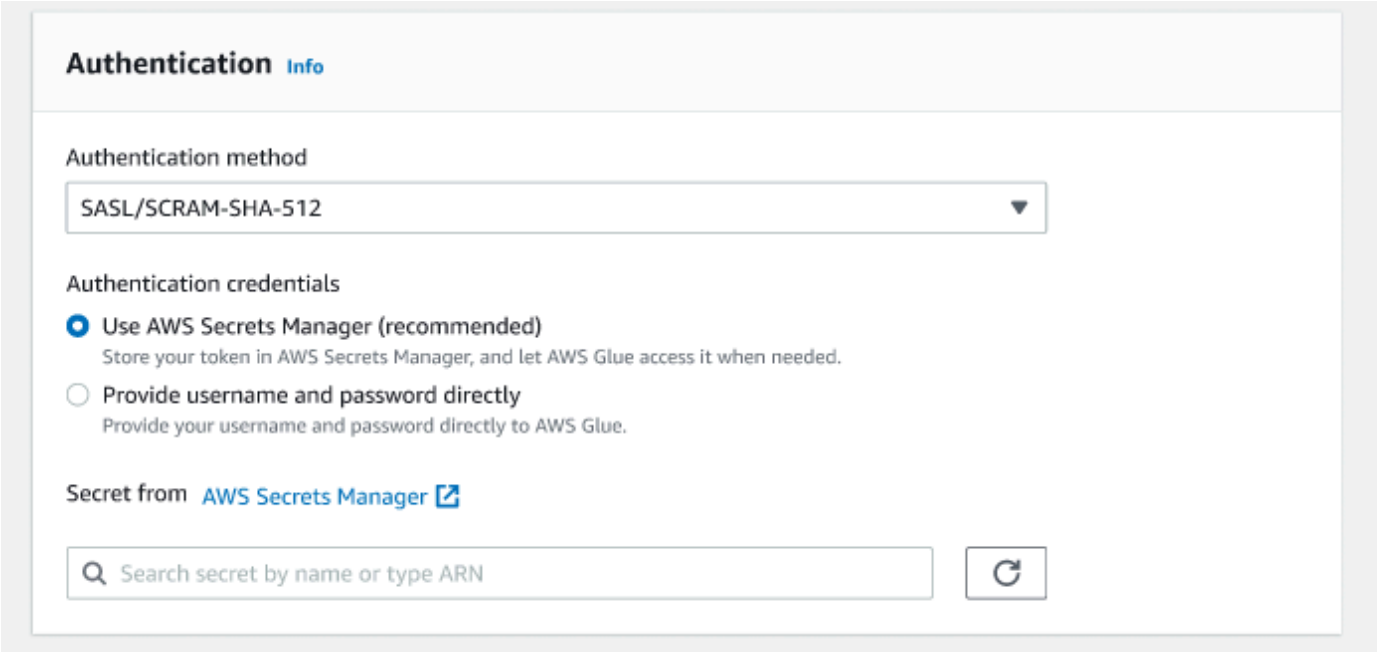
Authentifizierungsmethode auswählen



AWS Glue unterstützt das SASL-Framework (Simple Authentication and Security Layer) für die Authentifizierung. Das SASL-Framework unterstützt verschiedene Authentifizierungsmechanismen und AWS Glue bietet die Protokolle SCRAM (Benutzername und Passwort), GSSAPI (Kerberos-Protokoll) und PLAIN (Benutzername und Passwort).

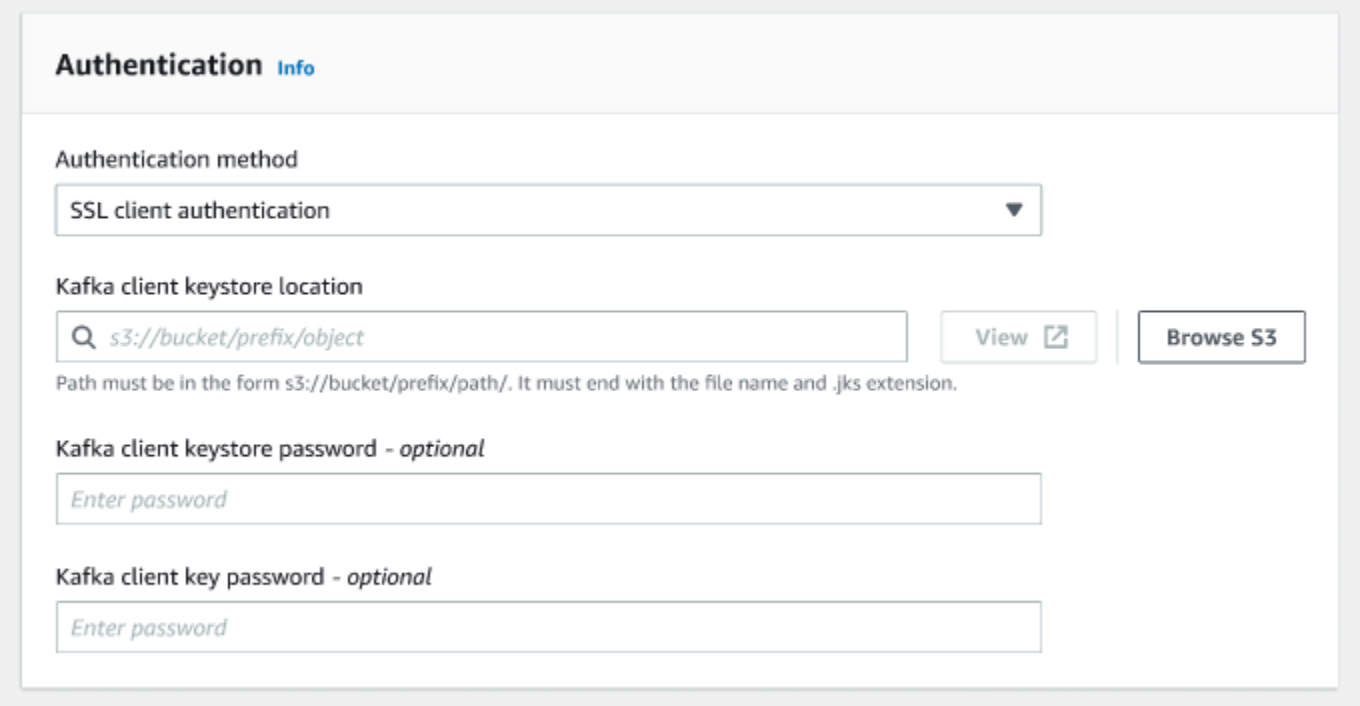
Wenn Sie eine Authentifizierungsmethode aus dem Dropdown-Menü auswählen, können die folgenden Client-Authentifizierungsmethoden ausgewählt werden:

- Keine – Keine Authentifizierung. Dies ist nützlich, wenn Sie eine Verbindung zu Testzwecken herstellen.
- SASL/SCRAM-SHA-512 – Wenn Sie diese Authentifizierungsmethode wählen, können Sie Anmeldeinformationen zur Authentifizierung angeben. Es gibt zwei Optionen:
  - AWS Secrets Manager verwenden (empfohlen) — Wenn Sie diese Option wählen, können Sie Ihre Anmeldeinformationen in AWS Secrets Manager speichern und bei Bedarf AWS Glue auf die Informationen zugreifen lassen. Geben Sie das Secret an, das die SSL- oder SASL-Authentifizierungsdaten speichert.



  - Geben Sie den Benutzernamen und das Passwort direkt ein.
- SASL/GSSAPI (Kerberos) - if you select this option, you can select the location of the keytab file, krb5.conf file and enter the Kerberos principal name and Kerberos service name. The locations for the keytab file and krb5.conf file must be in an Amazon S3 location. Since MSK does not yet support SASL/GSSAPI, diese Option ist nur für vom Kunden verwaltete Apache Kafka-Cluster verfügbar. Weitere Informationen finden Sie unter MIT Kerberos-Dokumentation: Keytab.
- SASL/PLAIN — Wählen Sie diese Authentifizierungsmethode, um Anmeldeinformationen anzugeben. Es gibt zwei Optionen:
  - AWS Secrets Manager verwenden (empfohlen) — Wenn Sie diese Option wählen, können Sie Ihre Anmeldeinformationen in AWS Secrets Manager speichern und bei Bedarf AWS Glue auf die Informationen zugreifen lassen. Geben Sie das Secret an, das die SSL- oder SASL-Authentifizierungsdaten speichert.
  - Geben Sie den Benutzernamen und das Passwort direkt ein.

- SSL-Clientauthentifizierung – Wenn Sie diese Option auswählen, können Sie den Standort des Kafka-Client-Keystores auswählen, indem Sie Amazon S3 durchsuchen. Optional können Sie das Kennwort für den Kafka-Client-Keystore und das Kafka-Client-Schlüsselkennwort eingeben.



Konfigurieren der Verschlüsselungs-Einstellungen

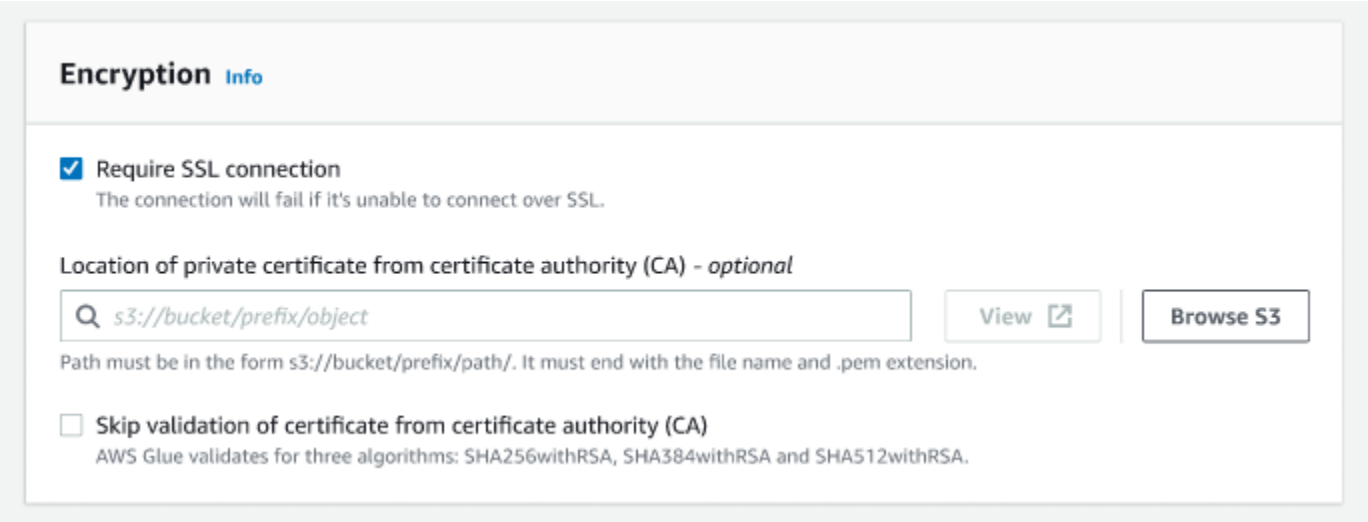
1. Wenn die Kafka-Verbindung eine SSL-Verbindung erfordert, aktivieren Sie das Kontrollkästchen für Require SSL connection (SSL-Verbindung erforderlich). Beachten Sie, dass die Verbindung fehlschlägt, wenn keine Verbindung über SSL herstellen werden kann. SSL für die Verschlüsselung kann mit jeder der Authentifizierungsmethoden (SASL/SCRAM-SHA-512, SASL/GSSAPI, SASL/PLAINoder der SSL-Client-Authentifizierung) verwendet werden und ist optional.

   Wenn die Authentifizierungsmethode auf SSL-Client-Authentifizierungfestgelegt ist, wird diese Option automatisch ausgewählt und deaktiviert, um Änderungen zu verhindern.

2. (Optional). Wählen Sie den Speicherort des privaten Zertifikats der Zertifizierungsstelle (CA). Beachten Sie, dass sich der Speicherort der Zertifizierung an einem S3-Standort befinden muss. Wählen Sie Browse (Durchsuchen), um die Datei aus einem verbundenen S3-Bucket auszuwählen. Der Pfad muss im Format `s3://bucket/prefix/filename.pem` eingegeben werden. Er muss mit dem Dateinamen und der Erweiterung .pem enden.

3. Sie können die Validierung des Zertifikats einer Zertifizierungsstelle (CA) überspringen. Wählen Sie das Kontrollkästchen Skip validation of certificate from certificate authority (CA) (Validierung des Zertifikats der Zertifizierungsstelle (CA). Wenn dieses Kästchen nicht aktiviert ist, validiert AWS Glue Zertifikate für drei Algorithmen:

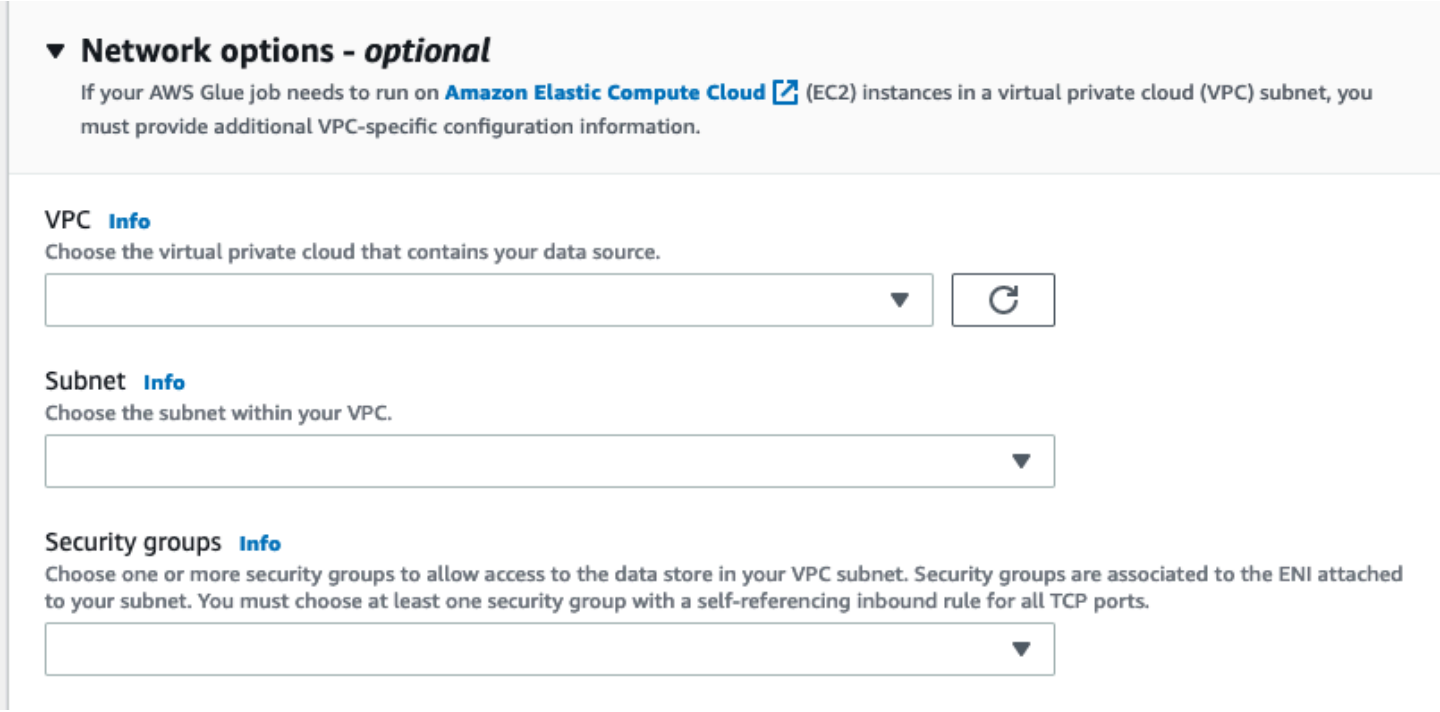
   - SHA256withRSA
   - SHA384withRSA
   - SHA512withRSA



(Optional) Netzwerkoptionen

Im Folgenden werden optionale Schritte zum Konfigurieren von VPC-, Subnetz- und Sicherheitsgruppen aufgeführt. Wenn Ihr AWS Glue Job auf EC2 Amazon-Instances in einem Virtual Private Cloud (VPC) -Subnetz ausgeführt werden muss, müssen Sie zusätzliche VPC-spezifische Konfigurationsinformationen angeben.

1. Wählen Sie die VPC (Virtual Private Cloud) aus, die Ihre Datenquelle enthält.
2. Wählen Sie das Subnetz mit Ihrer VPC aus.
3. Wählen Sie eine oder mehrere Sicherheitsgruppen aus, um den Zugriff auf den Datenspeicher in Ihrem VPC-Subnetz zuzulassen. Sicherheitsgruppen sind mit der Ihrem Subnetz angefügten ENI verknüpft. Sie müssen mindestens eine Sicherheitsgruppe mit einer selbstreferenzierenden eingehenden Regel für alle TCP-Ports auswählen.



Erstellen von Aufträgen mit benutzerdefinierten Connectors

Sie können Konnektoren und Verbindungen sowohl für Datenquellknoten als auch für Datenzielknoten verwenden in AWS Glue Studio.

Themen

- Erstellen von Aufträgen, die einen Connector für die Datenquelle verwenden
- Konfigurieren von Quelleneigenschaften für Knoten, die Connectors verwenden
- Konfigurieren von Zieleigenschaften für Knoten, die Connectors verwenden

Erstellen von Aufträgen, die einen Connector für die Datenquelle verwenden

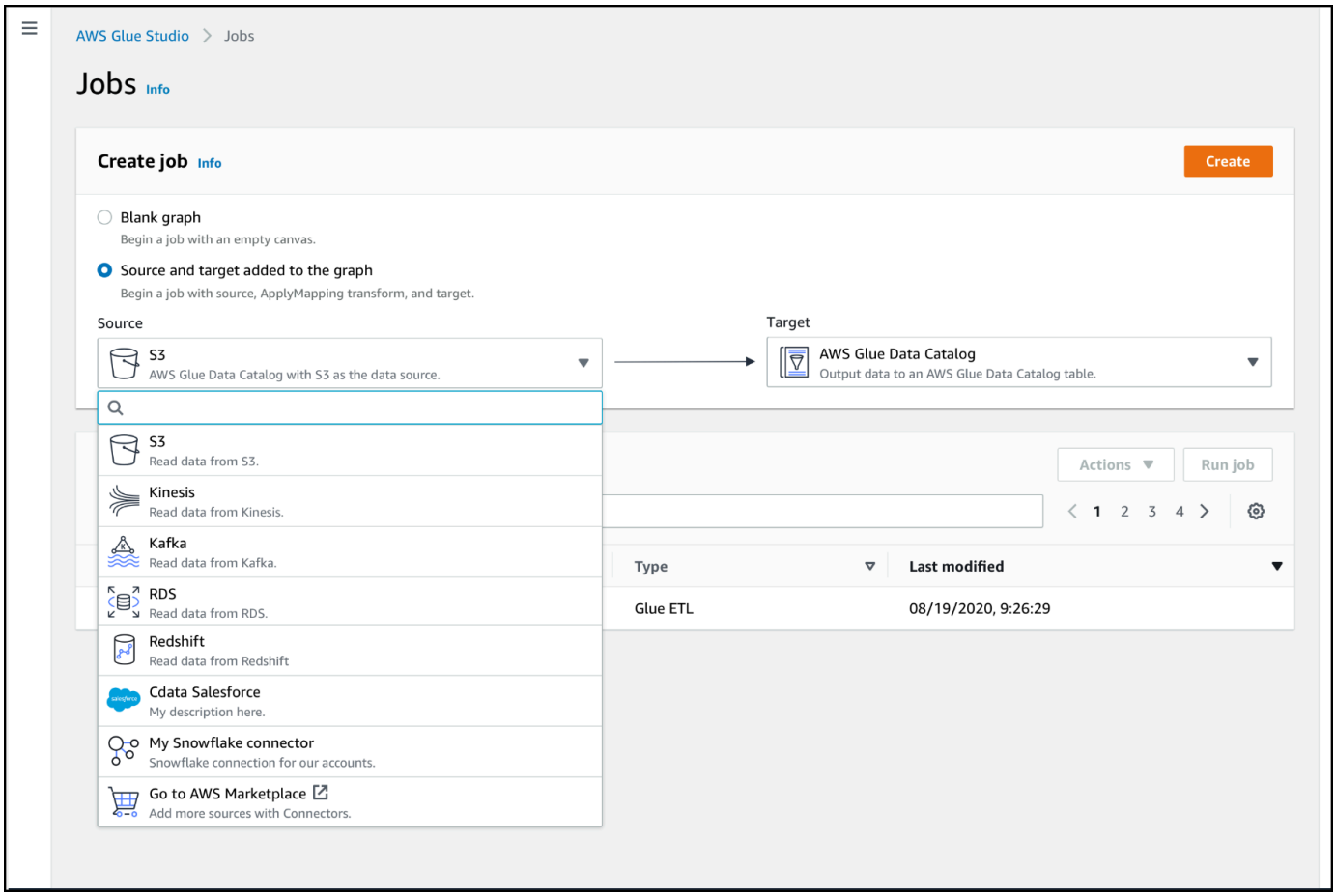
Wenn Sie einen neuen Auftrag erstellen, können Sie einen Connector für die Datenquelle und die Datenziele auswählen.

Aufträge erstellen, die Connectors für die Datenquelle oder das Datenziel verwenden

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Studio Konsole bei https://console.aws.amazon.com/gluestudio/.

2. Wählen Sie auf der Seite Connectors in der Ressourcenliste Your Connections (Ihre Verbindungen) die Verbindung aus, die Sie in Ihrem Auftrag verwenden möchten. Klicken Sie dann auf Create job (Auftrag erstellen).

   Alternativ auf dem AWS Glue Studio Wählen Sie auf der Seite Jobs unter Job erstellen die Option Quelle und Ziel aus, die dem Diagramm hinzugefügt wurden. In der Dropdown-Liste Source (Quelle) wählen Sie den benutzerdefinierten Connector aus, den Sie in Ihrem Auftrag verwenden möchten. Sie können auch einen Connector für Target (Ziel) auswählen.



3. Klicken Sie dann auf Create (Erstellen), um den visuellen Auftragseditor zu öffnen.
4. Konfigurieren Sie den Datenquellknoten, wie unter Konfigurieren von Quelleneigenschaften für Knoten, die Connectors verwenden beschrieben.
5. Fahren Sie mit der Erstellung Ihres ETL-Auftrags fort, indem Sie Transformationen, zusätzliche Datenspeicher und Datenziele hinzufügen, wie unter Starten von Visual ETL-Jobs in AWS Glue Studio beschrieben.
6. Passen Sie die Umgebung der Auftragsausführung an, indem Sie Auftragseigenschaften konfigurieren, wie unter Ändern der Auftragseigenschaften beschrieben.

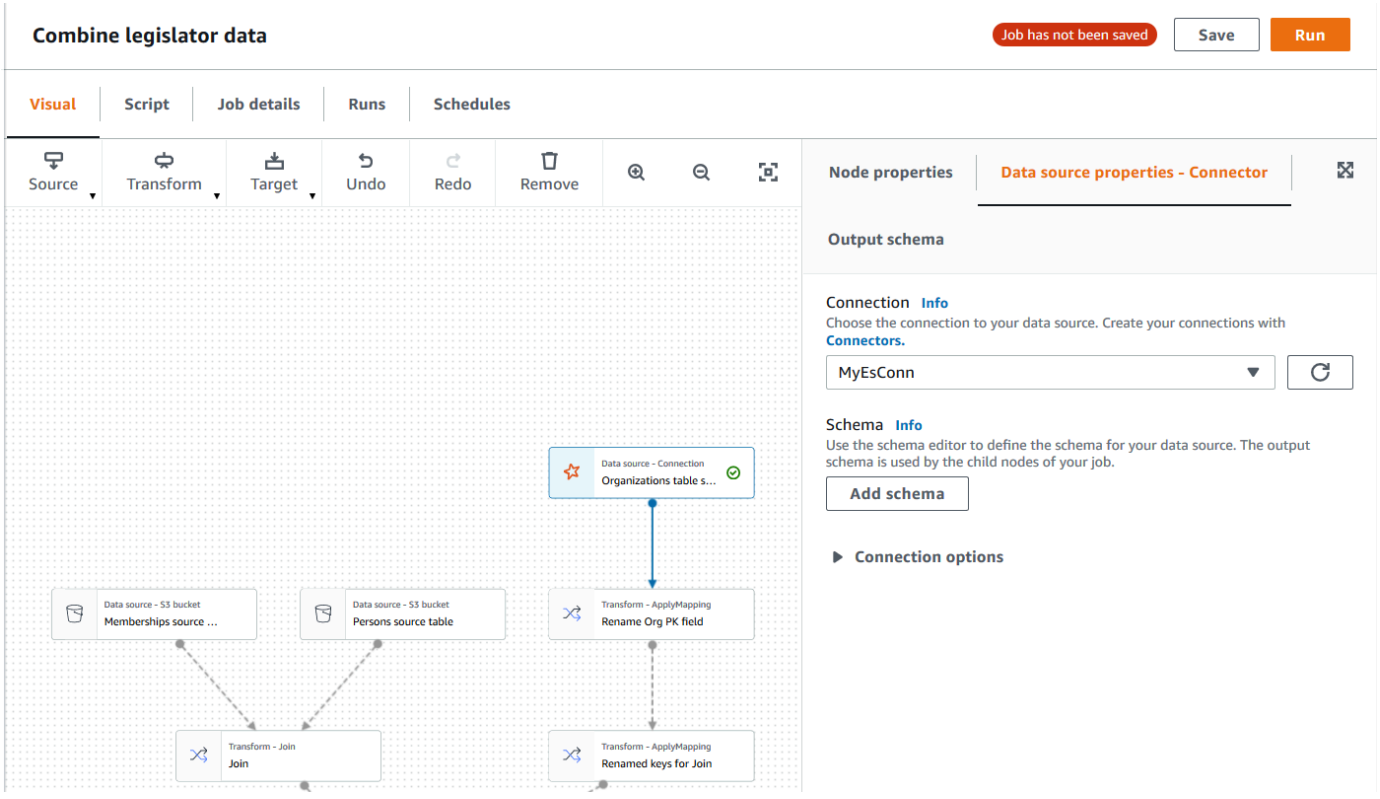
7. Speichern Sie den Auftrag und führen Sie ihn aus.

Konfigurieren von Quelleneigenschaften für Knoten, die Connectors verwenden

Nachdem Sie einen Auftrag erstellt haben, der einen Connector für die Datenquelle verwendet, zeigt der visuelle Auftragseditor ein Auftragsdiagramm mit einem Datenquellenknoten an, der für den Connector konfiguriert ist. Sie müssen die Datenquelleneigenschaften für diesen Knoten konfigurieren.

Eigenschaften für einen Datenquellenknoten konfigurieren, der einen Connector verwendet

1. Wählen Sie den Connector-Datenquellknoten im Auftragsdiagramm aus, oder fügen Sie einen neuen Knoten hinzu und wählen Sie den Connector für den Node type (Knotentyp) aus. Wählen Sie dann auf der rechten Seite im Bereich „Node Details“ (Knotendetails) die Registerkarte Data source properties (Datenquelleneigenschaften) aus, falls sie nicht bereits ausgewählt ist.



2. Wählen Sie auf der Registerkarte Data source properties (Datenquelleneigenschaften) die Verbindung aus, die Sie für diesen Auftrag verwenden möchten.

   Geben Sie die zusätzlichen Informationen ein, die für jeden Verbindungstyp erforderlich sind:

JDBC

- Data source input type (Eingabetyp der Datenquelle): Geben Sie entweder einen Tabellennamen oder eine SQL-Abfrage als Datenquelle an. Abhängig vom gewählten Typ müssen Sie die folgenden zusätzlichen Informationen eingeben:
  - Table name (Tabellenname): Der Name der Tabelle in der Datenquelle. Wenn die Datenquelle den Begriff Tabelle nicht verwendet, geben Sie den Namen einer geeigneten Datenstruktur an, wie in den Informationen zur Verwendung des benutzerdefinierten Konnektors angegeben (die unter verfügbar sind AWS Marketplace).
  - Filter predicate (Filterprädikat): Eine Bedingungsklausel, die beim Lesen der Datenquelle verwendet werden soll, ähnelt einer `WHERE`-Klausel zum Abrufen einer Teilmenge der Daten.
  - Query code (Abfragecode): Geben Sie eine SQL-Abfrage ein, die zum Abrufen eines bestimmten Datensatzes aus der Datenquelle verwendet werden soll. Ein Beispiel für eine einfache SQL-Abfrage:

    ```
    SELECT column_list FROM
                              table_name WHERE where_clause
    ```

- Schema: Weil AWS Glue Studio verwendet in der Verbindung gespeicherte Informationen für den Zugriff auf die Datenquelle, anstatt Metadateninformationen aus einer Datenkatalogtabelle abzurufen. Sie müssen die Schema-Metadaten für die Datenquelle angeben. Klicken Sie auf Add schema (Schema hinzufügen), um den Schema-Editor zu öffnen.

  Anweisungen zur Verwendung des Schema-Editors finden Sie unter Bearbeiten des Schemas in einem benutzerdefinierten Transformationsknoten.
- Partition column (Partitionsspalte): (Optional) Sie können die Datenlesevorgänge partitionieren, indem Sie Werte für Partition column (Partitionsspalte), Lower bound (Untergrenze), Upper bound (Obergrenze) und Number of partitions (Anzahl der Partitionen) aus.

  Die Werte für `lowerBound` und `upperBound` werden verwendet, um den Partitionsschritt zu bestimmen, nicht zum Filtern der Zeilen in der Tabelle. Alle Zeilen der Tabelle werden partitioniert und zurückgegeben.

> **Note**
>
> Die Spaltenpartitionierung fügt der Abfrage, die zum Lesen der Daten verwendet wird, eine zusätzliche Partitionierungsbedingung hinzu. Wenn Sie eine Abfrage anstelle eines Tabellennamens verwenden, sollten Sie überprüfen, ob die Abfrage mit der angegebenen Partitionierungsbedingung funktioniert. Zum Beispiel:
>
> - Wenn Ihr Abfrageformat `"SELECT col1 FROM table1"` lautet, dann testen Sie die Abfrage, indem Sie eine `WHERE`-Klausel am Ende der Abfrage stellen, die die Partitionsspalte verwendet.
> - Wenn Ihr Abfrageformat `"SELECT col1 FROM table1 WHERE col2=val"` lautet, dann testen Sie die Abfrage, indem Sie die `WHERE`-Klausel mit `AND` und einem Ausdruck erweitern, der die Partitionsspalte verwendet.

- Data type casting (Datentypumwandlung): Wenn die Datenquelle Datentypen verwendet, die in JDBC nicht verfügbar sind, geben Sie in diesem Abschnitt an, wie ein Datentyp aus der Datenquelle in JDBC-Datentypen konvertiert werden soll. Sie können bis zu 50 verschiedene Datentypkonvertierungen angeben. Alle Spalten in der Datenquelle, die denselben Datentyp verwenden, werden auf die gleiche Weise konvertiert.

  Wenn Sie beispielsweise über drei Spalten in der Datenquelle verfügen, die den Datentyp `Float` verwenden, und Sie angeben, dass der Datentyp `Float` in den JDBC–Datentyp `String` konvertiert werden soll, werden alle drei Spalten, die den Datentyp `Float` verwenden, in `String`-Datentypen konvertiert.

- Tasten für Job-Lesezeichen: Hilfe zu Job-Lesezeichen AWS Glue Behalten Sie Statusinformationen bei und verhindern Sie die Wiederverarbeitung alter Daten. Geben Sie eine oder mehrere Spalten als Lesezeichenschlüssel an. AWS Glue Studio verwendet Lesezeichenschlüssel, um Daten nachzuverfolgen, die bereits während einer früheren Ausführung des ETL-Jobs verarbeitet wurden. Alle Spalten, die Sie für benutzerdefinierte Lesezeichenschlüssel verwenden, müssen streng monoton erhöht oder verringert werden, aber Lücken sind zulässig.

  Wenn Sie mehrere Lesezeichenschlüssel eingeben, werden diese zu einem einzigen zusammengesetzten Schlüssel zusammengefasst. Ein zusammengesetzter Schlüssel für Auftragslesezeichen sollte keine doppelten Spalten enthalten. Wenn Sie keine Lesezeichenschlüssel angeben, AWS Glue Studio verwendet standardmäßig den Primärschlüssel als Lesezeichenschlüssel, vorausgesetzt, der Primärschlüssel wird

sequentiell erhöht oder verringert (ohne Lücken). Wenn die Tabelle keinen Primärschlüssel hat, aber die Eigenschaft „Job bookmark“ (Auftragslesezeichen) aktiviert ist, müssen Sie benutzerdefinierte Schlüssel für Auftragslesezeichen angeben. Andernfalls schlagen die Suche nach standardmäßig zu verwendenden Primärschlüsseln und die Auftragsausführung fehl.

- Job bookmark keys sorting order (Sortierreihenfolge der Schlüssel für Auftragslesezeichen): Wählen Sie aus, ob die Schlüsselwerte auf- oder absteigend sortiert werden.

Spark

- Schema: Weil AWS Glue Studio verwendet in der Verbindung gespeicherte Informationen für den Zugriff auf die Datenquelle, anstatt Metadateninformationen aus einer Datenkatalogtabelle abzurufen. Sie müssen die Schema-Metadaten für die Datenquelle angeben. Klicken Sie auf Add schema (Schema hinzufügen), um den Schema-Editor zu öffnen.

  Anweisungen zur Verwendung des Schema-Editors finden Sie unter Bearbeiten des Schemas in einem benutzerdefinierten Transformationsknoten.

- Connection options (Verbindungsoptionen): Geben Sie nach Bedarf weitere Schlüssel-Wert-Paare ein, um zusätzliche Verbindungsinformationen oder -optionen bereitzustellen. Sie können beispielsweise einen Datenbanknamen, einen Tabellennamen, einen Benutzernamen und ein Passwort eingeben.

  Beispielsweise geben Sie für OpenSearch die folgenden Schlüssel-Wert-Paare ein, wie unter beschrieben: the section called “ Tutorial: Den AWS Glue Connector für Elasticsearch verwenden ”

  - `es.net.http.auth.user` : *`username`*
  - `es.net.http.auth.pass` : *`password`*
  - `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
  - `es.port` : `443`
  - `path`: *`<Elasticsearch resource>`*
  - `es.nodes.wan.only` : `true`

Ein Beispiel für die mindestens zu verwendenden Verbindungsoptionen finden Sie im Beispieltestskript MinimalSparkConnectorTest.scala on GitHub, das die Verbindungsoptionen zeigt, die Sie normalerweise in einer Verbindung bereitstellen würden.

Athena

- Table name (Tabellenname): Der Name der Tabelle in der Datenquelle. Wenn Sie einen Konnektor zum Lesen aus CloudWatch Athena-Logs verwenden, würden Sie den Tabellennamen `all_log_streams` eingeben.
- Athena schema name (Name des Athena-Schemas): Wählen Sie das Schema in Ihrer Athena-Datenquelle aus, das der Datenbank entspricht, die die Tabelle enthält. Wenn Sie einen Konnektor zum Lesen aus CloudWatch Athena-Logs verwenden, würden Sie einen Schemanamen eingeben, der dem ähnelt`/aws/glue/`*`name`*.
- Schema: Weil AWS Glue Studio verwendet in der Verbindung gespeicherte Informationen für den Zugriff auf die Datenquelle, anstatt Metadateninformationen aus einer Datenkatalogtabelle abzurufen. Sie müssen die Schema-Metadaten für die Datenquelle angeben. Klicken Sie auf Add schema (Schema hinzufügen), um den Schema-Editor zu öffnen.

  Anweisungen zur Verwendung des Schema-Editors finden Sie unter Bearbeiten des Schemas in einem benutzerdefinierten Transformationsknoten.
- Additional connection options (Zusätzliche Verbindungsoptionen): Geben Sie nach Bedarf weitere Schlüssel-Wert-Paare ein, um zusätzliche Verbindungsinformationen oder -optionen bereitzustellen.

Ein Beispiel finden Sie in der `README.md` Datei unter https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/Athena. In den Schritten in diesem Dokument zeigt der Beispiel-Code die minimal erforderlichen Verbindungsoptionen `tableName`, `schemaName` und `className` an. Im Code-Beispiel werden diese Optionen als Teil der Variable `optionsMap` spezifiziert, aber Sie können sie für Ihre Verbindung angeben und dann die Verbindung verwenden.

3. (Optional) Nachdem Sie die erforderlichen Informationen angegeben haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das daraus resultierende Datenschema für die Datenquelle sehen. Das auf dieser Registerkarte angezeigte

Schema wird von allen untergeordneten Knoten verwendet, die Sie dem Auftragsdiagramm hinzufügen.

4. (Optional) Nachdem Sie die Knoteneigenschaften und Datenquelleneigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des Datensatzes aus Ihrer Datenquelle. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Konfigurieren von Zieleigenschaften für Knoten, die Connectors verwenden

Wenn Sie einen Connector für den Datenzieltyp verwenden, müssen Sie die Eigenschaften des Datenzielknotens konfigurieren.

Eigenschaften für einen Datenzielknoten konfigurieren, der einen Connector verwendet

1. Wählen Sie den Zielknoten der Connector-Daten im Auftragsdiagramm aus. Wählen Sie dann auf der rechten Seite im Bereich „Node Details“ (Knotendetails) die Registerkarte Data target properties (Datenzieleigenschaften) aus, falls sie nicht bereits ausgewählt ist.
2. Wählen Sie auf der Registerkarte Data target properties (Datenzieleigenschaften) die Verbindung für Schreibvorgänge im Ziel aus.

   Geben Sie die zusätzlichen Informationen ein, die für jeden Verbindungstyp erforderlich sind:

   JDBC

   - Connection (Verbindung): Wählen Sie die Verbindung aus, die Sie mit Ihrem Connector verwenden möchten. Weitere Informationen zum Herstellen einer Verbindung finden Sie unter Erstellen von Verbindungen für Connectors.
   - Table name (Tabellenname): Der Name der Tabelle im Datenziel. Wenn das Datenziel den Begriff Tabelle nicht verwendet, geben Sie den Namen einer geeigneten Datenstruktur an, wie in den Informationen zur Verwendung des benutzerdefinierten Konnektors angegeben (verfügbar unter AWS Marketplace).
   - Batch size (Batchgröße) (Optional): Geben Sie die Anzahl der Zeilen oder Datensätze ein, die in einem einzigen Vorgang in die Zieltabelle eingefügt werden sollen. Der Standardwert lautet 1 000.

Spark

- Connection (Verbindung): Wählen Sie die Verbindung aus, die Sie mit Ihrem Connector verwenden möchten. Wenn Sie zuvor keine Verbindung erstellt haben, wählen Sie Create connection (Verbindung erstellen) aus. Weitere Informationen zum Herstellen einer Verbindung finden Sie unter Erstellen von Verbindungen für Connectors.
- Connection options (Verbindungsoptionen): Geben Sie nach Bedarf weitere Schlüssel-Wert-Paare ein, um zusätzliche Verbindungsinformationen oder -optionen bereitzustellen. Sie können einen Datenbanknamen, einen Tabellennamen, einen Benutzernamen und ein Passwort eingeben.

  Beispielsweise geben Sie für OpenSearch die folgenden Schlüssel-Wert-Paare ein, wie unter beschrieben: the section called “ Tutorial: Den AWS Glue Connector für Elasticsearch verwenden ”

  - `es.net.http.auth.user` : *`username`*
  - `es.net.http.auth.pass` : *`password`*
  - `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
  - `es.port` : `443`
  - `path`: *`<Elasticsearch resource>`*
  - `es.nodes.wan.only` : `true`

  Ein Beispiel für die mindestens zu verwendenden Verbindungsoptionen finden Sie im Beispieltestskript MinimalSparkConnectorTest.scala on GitHub, das die Verbindungsoptionen zeigt, die Sie normalerweise in einer Verbindung bereitstellen würden.

3. Nachdem Sie die erforderlichen Informationen angegeben haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das daraus resultierende Datenschema für die Datenquelle sehen.

Verwalten von Connectors und Verbindungen

Sie verwenden die Seite Verbindungen in AWS Glue um Ihre Konnektoren und Verbindungen zu verwalten.

Themen

- Anzeigen von Connector- und Verbindungsdetails
- Bearbeiten von Connectors und Verbindungen
- Löschen von Connectors und Verbindungen
- Kündigen eines Abonnements eines Connectors

Anzeigen von Connector- und Verbindungsdetails

Eine Zusammenfassung zu Ihren Connectors und Verbindungen finden Sie in den Ressourcentabellen Your connectors (Ihre Connectors) und Your connections (Ihre Verbindungen) auf der Seite Connectors. Führen Sie die folgenden Schritte aus, um Details anzuzeigen.

Anschluss- oder Verbindungsdetails anzeigen

1. Im AWS Glue Studio Konsole, wählen Sie im Navigationsbereich der Konsole Connectors aus.
2. Wählen Sie den Connector oder die Verbindung aus, für den bzw. die Sie detaillierte Informationen anzeigen möchten.
3. Klicken Sie auf Actions (Aktionen) und danach auf View details (Details anzeigen), um die Seite mit Details zu dem Connector bzw. der Verbindung zu öffnen.
4. Auf der Detailseite können Sie für den Connector bzw. die Verbindung Edit (Bearbeiten) oder Delete (Löschen) auswählen.

    - Bei Connectors können Sie mit Create Connection (Verbindung erstellen) eine neue Verbindung erstellen, die diesen Connector verwendet.
    - Bei Verbindungen können Sie mit Create job (Auftrag erstellen) einen Auftrag erstellen, der die Verbindung verwendet.

Bearbeiten von Connectors und Verbindungen

Auf der Seite Connectors können Sie in Ihren Connectors bzw. Verbindungen gespeicherten Informationen ändern.

Connectors oder Verbindungen bearbeiten

1. Im AWS Glue Studio Konsole, wählen Sie im Navigationsbereich der Konsole Connectors aus.
2. Wählen Sie den Connector/die Verbindung aus, den/die Sie bearbeiten möchten.
3. Wählen Sie Actions und anschließend Bearbeiten.

Sie können auch auf View details (Details anzeigen) klicken und auf der Seite mit Details zum Connector bzw. zur Verbindung Edit (Bearbeiten) auswählen.

4. Aktualisieren Sie die Informationen auf der Seite Edit connector (Connector bearbeiten) bzw. Edit connection (Verbindung bearbeiten) und klicken Sie auf Save (Speichern).

Löschen von Connectors und Verbindungen

Auf der Seite Connectors können Sie Connectors und Verbindungen löschen. Wenn Sie einen Connector löschen, sollten alle für ihn erstellten Verbindungen ebenfalls gelöscht werden.

Um Konnektoren zu entfernen von AWS Glue Studio

1. Im AWS Glue Studio Konsole: Wählen Sie im Navigationsbereich der Konsole Connectors aus.
2. Wählen Sie den Connector/die Verbindung aus, den/die Sie löschen möchten.
3. Wählen Sie Actions (Aktionen) und anschließend Delete (Löschen) aus.

   Sie können auch auf View details (Details anzeigen) klicken und auf der Seite mit Details zum Connector bzw. zur Verbindung Delete (Löschen) auswählen.
4. Bestätigen Sie, dass Sie den Connector oder die Verbindung entfernen möchten, indem Sie **`Delete`** eingeben und dann Delete (Löschen) auswählen.

   Wenn Sie einen Connector löschen, werden alle für ihn erstellten Verbindungen ebenfalls gelöscht.

Aufträge, die eine gelöschte Verbindung verwenden, funktionieren nicht mehr. Sie können die Aufträge entweder bearbeiten, um einen anderen Datenspeicher zu verwenden, oder die Aufträge entfernen. Informationen zum Löschen von Aufträgen finden Sie unter Löschen von Aufträgen.

Wenn Sie einen Connector löschen, wird das Abonnement für den Connector in AWS Marketplace nicht gekündigt. Um ein Abonnement für einen gelöschten Connector zu entfernen, folgen Sie den Anweisungen unterKündigen eines Abonnements eines Connectors.

Kündigen eines Abonnements eines Connectors

Nachdem Sie die Verbindungen und den Connector von gelöscht haben AWS Glue Studio, können Sie Ihr Abonnement kündigen, AWS Marketplace wenn Sie den Connector nicht mehr benötigen.

> **Note**
>
> Wenn Sie Ihr Abonnement für einen Connector kündigen, wird der Connector oder die Verbindung nicht von Ihrem Konto entfernt. Alle Aufträge, die den Connector und die zugehörigen Verbindungen verwenden, können den Connector nicht mehr verwenden und schlagen fehl.
> Bevor Sie einen Connector abbestellen oder erneut abonnieren AWS Marketplace, sollten Sie bestehende Verbindungen und Connectors löschen, die mit diesem AWS Marketplace Produkt verknüpft sind.

Um sich von einem Connector abzumelden, finden Sie unter AWS Marketplace

1. Melden Sie sich bei der AWS Marketplace Konsole unter https://console.aws.amazon.com/marketplace an.
2. Wählen Sie Manage subscriptions (Abonnements verwalten).
3. Klicken Sie auf der Seite Manage subscriptions (Abonnements verwalten) auf Manage (Verwalten) neben dem Connector, dessen Abonnement Sie kündigen möchten.
4. Wählen Sie anschließend Actions (Aktionen) und Cancel Subscription (Abonnement kündigen) aus.
5. Aktivieren Sie das Kontrollkästchen, um zu bestätigen, dass ausgeführte Instances auf Ihrem Konto abgerechnet werden, und wählen Sie dann Yes, cancel subscription (Ja, Abonnement kündigen) aus.

Entwickeln von benutzerdefinierten Connectors

Sie können den Code schreiben, der Daten aus Ihrem Datenspeicher liest oder in diesen schreibt und die Daten für die Verwendung mit formatiert AWS Glue Studio Jobs. Sie können Connectors für Spark, Athena und JDBC-Datenspeicher erstellen. Der auf veröffentlichte Beispielcode GitHub bietet einen Überblick über die grundlegenden Schnittstellen, die Sie implementieren müssen.

Sie benötigen eine lokale Entwicklungsumgebung, um Ihren Connector-Code zu erstellen. Sie können eine beliebige IDE oder sogar nur einen Befehlszeileneditor verwenden, um Ihren Connector zu schreiben. Beispiele für Entwicklungsumgebungen:

- Eine lokale Scala-Umgebung mit einer lokalen AWS Glue ETL-Maven-Bibliothek, wie unter Lokales Entwickeln mit Scala im Entwicklerhandbuch beschrieben.AWS Glue

- IntelliJ IDE, indem Sie die IDE von herunterladen. https://www.jetbrains.com/idea/

Themen

- Entwickeln von Spark-Connectors
- Entwickeln von Athena-Connectors
- Entwickeln von JDBC-Connectors
- Beispiele für die Verwendung benutzerdefinierter Konnektoren mit AWS Glue Studio
- Entwickeln AWS Glue Steckverbinder für AWS Marketplace

Entwickeln von Spark-Connectors

Sie können mit Spark DataSource API V2 (Spark 2.4) einen Spark-Konnektor zum Lesen von Daten erstellen.

Erstellen Sie einen benutzerdefinierten Spark-Connector wie folgt

Folgen Sie den Schritten in AWS Glue GitHub Beispielbibliothek für die Entwicklung von Spark-Konnektoren, die sich unter https://github.com/aws-samples/aws-glue-samplestree/master/GlueCustomConnectors/development/Spark/README/.md befindet.

Entwickeln von Athena-Connectors

Sie können einen Athena-Konnektor erstellen, der verwendet werden kann von AWS Glue and AWS Glue Studio um eine benutzerdefinierte Datenquelle abzufragen.

Erstellen Sie einen benutzerdefinierten Athena-Connector wie folgt

Folgen Sie den Schritten in AWS Glue GitHub Beispielbibliothek für die Entwicklung von Athena-Konnektoren, die sich unter https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/Athenabefindet.

Entwickeln von JDBC-Connectors

Sie können einen Connector erstellen, der JDBC verwendet, um auf Ihre Datenspeicher zuzugreifen.

Einen benutzerdefinierten JDBC-Connector erstellen

1. Installieren Sie das AWS Glue Spark-Laufzeitbibliotheken in Ihrer lokalen Entwicklungsumgebung. Weitere Informationen finden Sie in der AWS Glue GitHub

Beispielbibliothek unter https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/GlueSparkRuntime/README.md.

2. Implementieren Sie den JDBC-Treiber, der für das Abrufen der Daten aus der Datenquelle verantwortlich ist. Weitere Informationen finden Sie in der Java-Dokumentation für Java SE 8.

   Erstellen Sie einen Einstiegspunkt in Ihrem Code, der AWS Glue Studio verwendet, um Ihren Connector zu finden. Das Feld Class name (Klassenname) sollte der vollständige Pfad Ihres JDBC-Treibers sein.

3. Verwenden Sie die `GlueContext`-API, um Daten mit dem Connector zu lesen. Benutzer können weitere Eingabeoptionen in der AWS Glue Studio Konsole, um bei Bedarf die Verbindung zur Datenquelle zu konfigurieren. Ein Codebeispiel, das zeigt, wie mit einem benutzerdefinierten JDBC-Connector aus einer JDBC-Datenbank gelesen und in eine JDBC-Datenbank geschrieben wird, finden Sie unter Custom- und ConnectionType-Werte. AWS Marketplace

Beispiele für die Verwendung benutzerdefinierter Konnektoren mit AWS Glue Studio

Beispiele für die Verwendung von benutzerdefinierten Connectors finden Sie in den folgenden Blogs:

- Entwickeln, Testen und Bereitstellen von benutzerdefinierten Konnektoren für Ihre Datenspeicher mit AWS Glue
- Apache Hudi: Schreiben in Apache Hudi-Tabellen mit AWS Glue Benutzerdefinierter Konnektor
- Google BigQuery: Migrieren von Daten von Google BigQuery zu Amazon S3 mit AWS Glue benutzerdefinierte Konnektoren
- Snowflake (JDBC): Durchführen von Datentransformationen mit Snowflake und AWS Glue
- SingleStore: Schnelles SingleStore ETL erstellen mit und AWS Glue
- Salesforce: Investieren Sie Salesforce-Daten mithilfe des benutzerdefinierten CData JDBC-Connectors in Amazon S3 mit - AWS Glue
- MongoDB: Gebäude AWS Glue Spark ETL-Jobs mit Amazon DocumentDB (mit MongoDB-Kompatibilität) und MongoDB
- Amazon Relational Database Service (Amazon RDS): Gebäude AWS Glue Entfachen Sie ETL-Jobs, indem Sie Ihre eigenen JDBC-Treiber für Amazon RDS mitbringen
- MySQL (JDBC): https://github.com/aws-samples/aws-glue-samples/.scala blob/master/GlueCustomConnectors/development/Spark/SparkConnectorMySQL

Entwickeln AWS Glue Steckverbinder für AWS Marketplace

Als AWS Partner können Sie benutzerdefinierte Konnektoren erstellen und diese hochladen, um sie AWS Marketplace an zu verkaufen AWS Glue Kunden.

Der Prozess zum Entwickeln des Connector-Codes ist der gleiche wie für benutzerdefinierte Connectors, aber der Prozess zum Hochladen und Überprüfen des Connector-Codes ist umfassender. Weitere Informationen finden Sie in den Anweisungen unter Konnektoren erstellen für AWS Marketplace auf der GitHub Website.

Einschränkungen für die Verwendung von Anschlüssen und Verbindungen in AWS Glue Studio

Wenn Sie benutzerdefinierte Konnektoren oder Konnektoren von verwenden AWS Marketplace, beachten Sie die folgenden Einschränkungen:

- Die testConnection-API wird bei Verbindungen, die für benutzerdefinierte Connectors erstellt wurden, nicht unterstützt.
- Die Passwortverschlüsselung der Data-Catalog-Verbindung wird von benutzerdefinierten Connectors nicht unterstützt.
- Sie können keine Auftragslesezeichen verwenden, wenn Sie ein Filterprädikat für einen Datenquellenknoten angeben, der einen JDBC-Connector verwendet.
- Das Erstellen einer Marketplace-Verbindung wird außerhalb der AWS Glue Studio Benutzeroberfläche nicht unterstützt.

Testen eines AWS Glue Verbindung

Es hat sich bewährt, vor der Verwendung eines AWS Glue Verwenden Sie für die Verbindung in einem ETL-Job den AWS Glue Konsole, um die Verbindung zu testen. AWS Glue verwendet die Parameter in Ihrer Verbindung, um zu bestätigen, dass sie auf Ihren Datenspeicher zugreifen kann, und meldet alle Fehler. Für Informationen über AWS Glue Verbindungen finden Sie unterHerstellen einer Verbindung zu Daten.

Um ein zu testen AWS Glue Verbindung

1. Melden Sie sich bei an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.

2. Wählen Sie im Navigationsbereich unter Data Catalog die Option Verbindungen aus. Sie können im Navigationsbereich auch Datenverbindungen oberhalb von Data Catalog auswählen.
3. Aktivieren Sie unter Verbindungen das Kontrollkästchen neben der gewünschten Verbindung und wählen Sie dann Aktionen aus. Wählen Sie im Dropdown-Menü die Option Verbindung testen aus.
4. Wählen Sie im Dialogfeld Verbindung testen eine Rolle aus, oder wählen Sie IAM-Rolle erstellen, um zur AWS Identity and Access Management (IAM-) Konsole zu wechseln und eine neue Rolle zu erstellen. Die Rolle muss über Berechtigungen für den Datenspeicher verfügen.
5. Wählen Sie Bestätigen aus.

   Der Test beginnt und kann einige Minuten dauern. Wenn der Test fehlschlägt, wählen Sie Fehlerbehebung, um die Schritte zur Behebung des Problems anzuzeigen.
6. Wählen Sie Logs aus, um die Logs einzusehen. CloudWatch Sie müssen über die erforderlichen IAM-Berechtigungen zum Anzeigen der Protokolle verfügen. Weitere Informationen finden Sie unter AWS Verwaltete (vordefinierte) Richtlinien für CloudWatch Protokolle im Amazon CloudWatch Logs-Benutzerhandbuch.

Konfiguration von AWS Anrufen für die Weiterleitung über Ihre VPC

Mit dem speziellen Jobparameter `disable-proxy-v2` können Sie Ihre Anrufe an Dienste wie Amazon S3 und AWS Glue über Ihre VPC weiterleiten. CloudWatch AWS Glue Verwendet standardmäßig einen lokalen Proxy, um Traffic über die AWS Glue VPC zu senden, um Skripte und Bibliotheken von Amazon S3 herunterzuladen, Anfragen zur Veröffentlichung von Protokollen und Metriken an zu senden und Anfragen AWS Glue für den Zugriff auf Datenkataloge zu senden. CloudWatch Dieser Proxy ermöglicht es, dass der Job normal funktioniert, auch wenn Ihre VPC keine richtige Route zu anderen AWS Diensten wie Amazon S3 CloudWatch, und AWS Glue konfiguriert. AWS Glue bietet jetzt einen Parameter, mit dem Sie dieses Verhalten ausschalten können. Weitere Informationen finden Sie unter Job-Parameter verwendet von AWS Glue. AWS Glue wird weiterhin den lokalen Proxy für die Veröffentlichung von CloudWatch Protokollen Ihrer AWS Glue Jobs verwenden.

> **Note**
>
> - Diese Funktion wird für AWS Glue Jobs mit AWS Glue Version 2.0 und höher unterstützt. Wenn Sie dieses Feature verwenden, müssen Sie sicherstellen, dass für Ihre VPC eine Route zu Amazon S3 über einen NAT- oder Service-VPC-Endpunkt konfiguriert ist.

- Der veraltete Auftrags-Parameter `disable-proxy` leitet Ihre Anrufe nur an Amazon S3 weiter, um Skripte und Bibliotheken über Ihre VPC herunterzuladen. Es wird empfohlen, stattdessen den neuen `disable-proxy-v2`-Parameter zu verwenden.

Beispielverwendung

Erstellen Sie einen AWS Glue Job mit`disable-proxy-v2`:

```
aws glue create-job \
    --name no-proxy-job \
    --role GlueDefaultRole \
    --command "Name=glueetl,ScriptLocation=s3://my-bucket/glue-script.py" \
    --connections Connections="traffic-monitored-connection" \
    --default-arguments '{"--disable-proxy-v2" : "true"}'
```

Herstellen einer Verbindung mit einem JDBC-Datenspeicher in einer VPC

Typischerweise erstellen Sie diese Ressourcen innerhalb von Amazon Virtual Private Cloud (Amazon VPC), so dass sie nicht über das öffentliche Internet zugänglich sind. Standardmäßig AWS Glue kann nicht auf Ressourcen innerhalb einer VPC zugegriffen werden. Um den Zugriff auf Ressourcen innerhalb Ihrer VPC AWS Glue zu ermöglichen, müssen Sie zusätzliche VPC-spezifische Konfigurationsinformationen angeben, die das IDs VPC-Subnetz und die Sicherheitsgruppe umfassen. IDs AWS Glue verwendet diese Informationen, um elastische Netzwerkschnittstellen einzurichten, die es Ihrer Funktion ermöglichen, eine sichere Verbindung zu anderen Ressourcen in Ihrer privaten VPC herzustellen.

Wenn Sie einen VPC-Endpunkt verwenden, fügen Sie ihn zu Ihrer Routing-Tabelle hinzu. Weitere Informationen finden Sie unter Erstellen eines Schnittstellen-VPC-Endpunkts für AWS Glue und Voraussetzungen.

Wenn Sie die Verschlüsselung in Data Catalog verwenden, erstellen Sie den Endpunkt der Benutzeroberfläche von KMS und fügen ihn zu Ihrer Routing-Tabelle hinzu. Weitere Informationen finden Sie unter Erstellen eines VPC-Endpunkts für AWS KMS.

Zugriff auf VPC-Daten über Elastic Network-Schnittstellen

Wenn eine AWS Glue Verbindung zu einem JDBC-Datenspeicher in einer VPC hergestellt AWS Glue wird, wird in Ihrem Konto eine elastic network interface (mit dem Präfix`Glue_`) für den Zugriff auf Ihre VPC-Daten erstellt. Sie können diese Netzwerkschnittstelle nicht löschen, solange sie angeschlossen ist. AWS Glue Ordnet im Rahmen der Erstellung der elastic network interface eine oder mehrere Sicherheitsgruppen zu. AWS Glue Um die Erstellung der Netzwerkschnittstelle AWS Glue zu ermöglichen, müssen Sicherheitsgruppen, die der Ressource zugeordnet sind, eingehenden Zugriff mit einer Quellregel zulassen. Diese Regel enthält eine Sicherheitsgruppe, die mit der Ressource verknüpft ist. Dies bietet der Elastic Network-Schnittstelle Zugriff auf Ihren Datenspeicher mit der gleichen Sicherheitsgruppe.

Um die Kommunikation mit ihren Komponenten AWS Glue zu ermöglichen, geben Sie eine Sicherheitsgruppe mit einer selbstreferenzierenden Regel für eingehenden Datenverkehr für alle TCP-Ports an. Durch Erstellen einer selbstreferenzierenden Regel können Sie die Quelle auf die gleiche Sicherheitsgruppe in der VPC beschränken und vermeiden, dass sie für alle Netzwerke geöffnet wird. Die Standardsicherheitsgruppe für Ihre VPC verfügt möglicherweise bereits über eine selbstreferenzierende eingehende Regel für `ALL Traffic`.

Sie können Regeln in der Amazon VPC-Konsole erstellen. Um die Regeleinstellungen über die AWS Management Console zu aktualisieren, navigieren Sie zur VPC-Konsole (https://console.aws.amazon.com/vpc/) und wählen Sie die entsprechende Sicherheitsgruppe aus. Geben Sie die eingehende Regel für `ALL TCP` an, damit sie als Quelle denselben Sicherheitsgruppennamen hat. Weitere Informationen zu Sicherheitsgruppenregeln finden Sie unter Sicherheitsgruppen für Ihre VPC.

Jeder Elastic Network-Schnittstelle wird eine private IP-Adresse aus dem IP-Adressbereich innerhalb der von Ihnen angegebenen Subnetze zugeordnet. Der Netzwerkschnittstelle sind keine öffentlichen IP-Adressen zugewiesen. AWS Glue erfordert Internetzugang (z. B. für den Zugriff auf AWS Dienste, die keine VPC-Endpunkte haben). Sie können eine NAT-Instance (Network Address Translation) in Ihrer VPC erstellen oder das NAT-Gateway von Amazon VPC verwenden. Weitere Informationen finden Sie unter NAT-Gateways im Amazon-VPC-Benutzerhandbuch. Sie können nicht direkt ein Internet-Gateway als Route in Ihrer Subnetzroutentabelle verwenden, das an Ihre VPC angefügt ist, da dafür die Netzwerkschnittstelle über öffentliche IP-Adressen verfügen muss.

Die VPC-Netzwerkattribute `enableDnsHostnames` und `enableDnsSupport` müssen auf "true" gesetzt werden. Weitere Informationen finden Sie unter Verwenden von DNS in Ihrer VPC.

> **Important**
>
> Richten Sie Ihren Datenspeicher nicht in einem öffentlichen Subnetz oder in einem privaten Subnetz ein, das über keinen Internetzugang verfügt. Binden Sie sie stattdessen ausschließlich an private Subnetze an, die über eine NAT-Instance oder ein Amazon-VPC-NAT-Gateway auf das Internet zugreifen.

Eigenschaften der Elastic Network-Schnittstelle

Um die Elastic Network-Schnittstelle zu erstellen, müssen Sie die folgenden Eigenschaften angeben:

VPC

Der Name der VPC, die Ihren Datenspeicher enthält.

Subnetz

Das Subnetz in der VPC, die den Datenspeicher enthält.

Sicherheitsgruppen

Die Sicherheitsgruppen, die mit dem Datenspeicher verknüpft sind. AWS Glue ordnet diese Sicherheitsgruppen der Elastic Network-Schnittstelle zu, die mit Ihrem VPC-Subnetz verbunden ist. Damit AWS Glue -Komponenten kommunizieren können und den Zugriff von anderen Netzwerken verhindern, muss mindestens eine ausgewählte Sicherheitsgruppe eine selbstreferenzierende Regel für eingehenden Datenverkehr für alle TCP-Ports angeben.

Weitere Informationen zur Verwaltung einer VPC mit Amazon Redshift finden Sie im Artikel zum Verwalten von Clustern in einer Amazon Virtual Private Cloud (VPC).

Weitere Informationen zum Verwalten einer VPC mit Amazon Relational Database Service (Amazon RDS) finden Sie unter Working with an Amazon RDS DB Instance in a VPC (Arbeiten mit einer Amazon-RDS-DB-Instance in einer VPC).

Verwenden einer MongoDB- oder MongoDB-Atlas-Verbindung

Nachdem Sie eine Verbindung für MongoDB oder MongoDB Atlas erstellt haben, können Sie diese Verbindung in Ihrem ETL-Auftrag verwenden. Sie erstellen eine Tabelle in der AWS Glue Data

Catalog und geben die MongoDB- oder MongoDB-Atlas-Verbindung für das `connection` Attribut der Tabelle an.

AWS Glue speichert Ihre Verbindung `url` und Ihre Anmeldeinformationen in der MongoDB-Verbindung. Verbindungs-URI-Formate sind wie folgt:

- Für MongoDB: mongodb://host:port/database. Der Host kann ein Hostname, eine IP-Adresse oder ein UNIX-Domain-Socket sein. Wenn die Verbindungszeichenfolge keinen Port angibt, wird der standardmäßige MongoDB-Port 27017 verwendet.
- Für MongoDB Atlas: mongodb+srv://server.example.com/database. Der Host kann ein Hostname sein, der im Folgenden einem DNS-SRV-Eintrag entspricht. Das SRV-Format benötigt keinen Port und verwendet den standardmäßigen MongoDB-Port 27017.

Darüber hinaus können Sie Optionen in Ihrem Auftragsskript angeben. Weitere Informationen finden Sie unter the section called “MongoDB-Verbindung”.

Crawling eines Amazon-S3-Datenspeichers mit einem VPC-Endpunkt

Zu Sicherheits-, Überwachungs- oder Kontrollzwecken möchten Sie möglicherweise, dass auf Ihren Amazon-S3-Datenspeicher oder Ihre Amazon-S3-gestützte Datenkatalog-Tabelle nur über eine Amazon-Virtual-Private-Cloud(Amazon VPC)-Umgebung zugegriffen wird. In diesem Thema wird beschrieben, wie Sie eine Verbindung zum Amazon-S3-Datenspeicher oder zur Amazon-S3-gestützten Datenkatalog-Tabelle in einem VPC-Endpunkt mithilfe der `Network`-Verbindung herstellen.

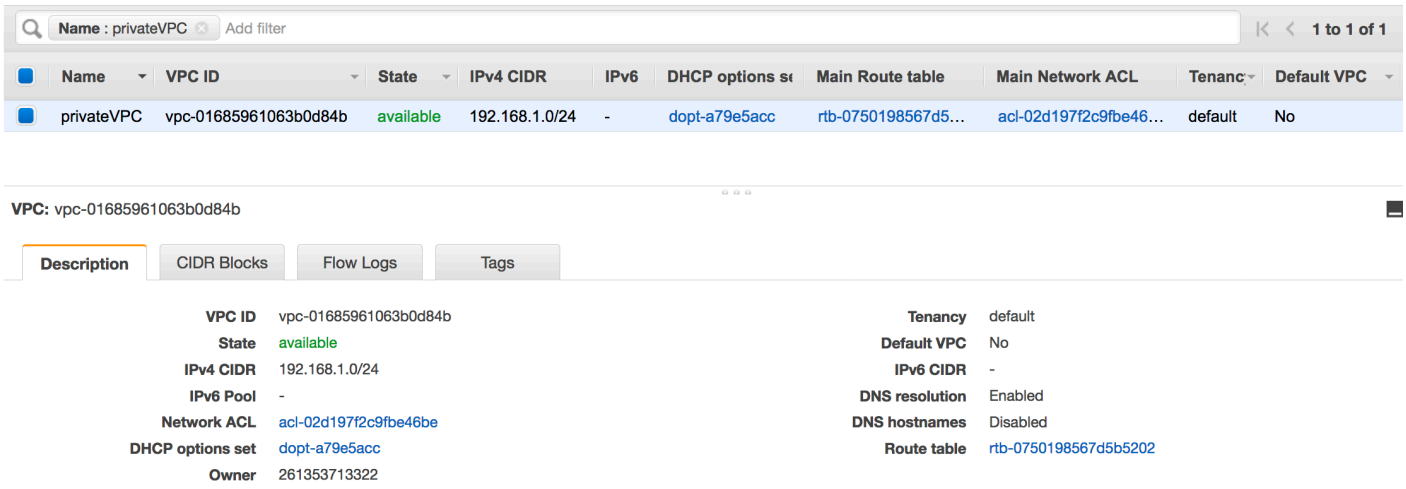
Führen Sie die folgenden Aufgaben aus, um einen Crawler im Datenspeicher auszuführen:

- the section called “Voraussetzungen”
- the section called “Herstellen der Verbindung zu Amazon S3”
- the section called “Testen der Verbindung zu Amazon S3”
- the section called “Erstellen eines Crawlers für einen Amazon-S3-Datenspeicher”
- the section called “Ausführen eines Crawlers”

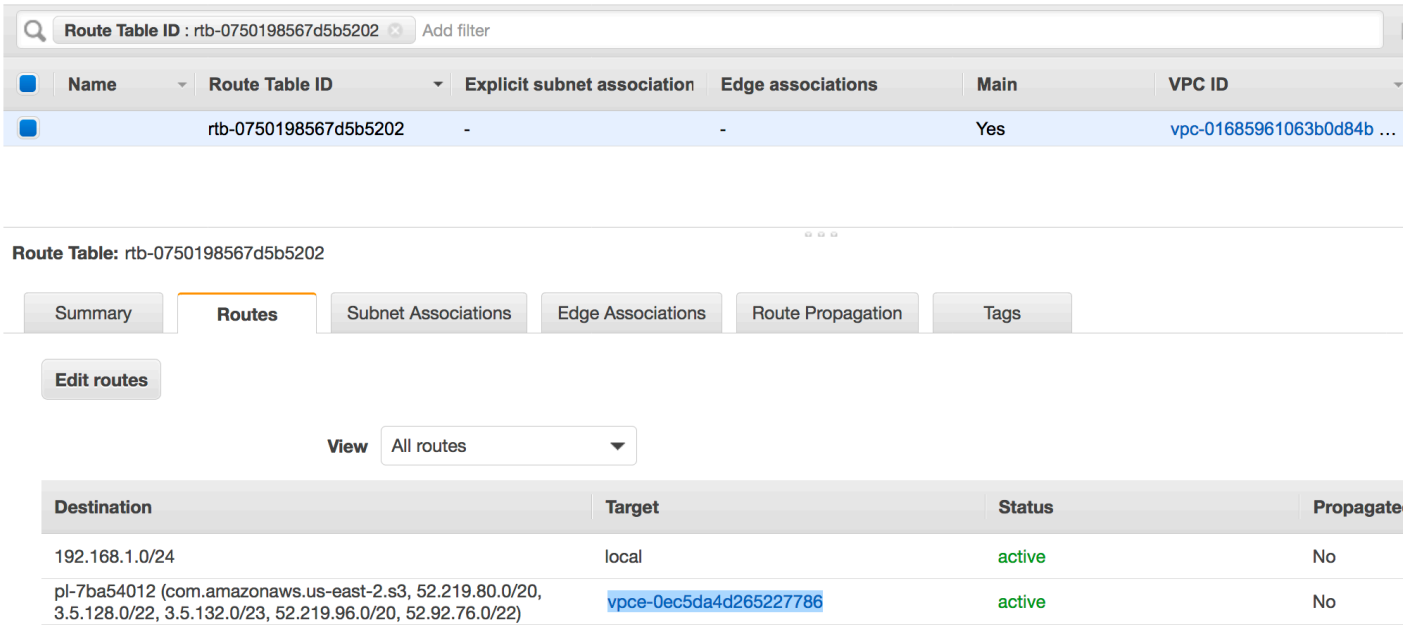
Voraussetzungen

Überprüfen Sie, ob Sie diese Voraussetzungen für die Einrichtung Ihres Amazon-S3-Datenspeichers oder Ihrer Amazon-S3-gestützten Datenkatalog-Tabelle für den Zugriff über eine Amazon-Virtual-Private-Cloud(Amazon VPC)-Umgebung erfüllt haben.

- Eine konfigurierte VPC. Zum Beispiel: vpc-01685961063b0d84b. Weitere Informationen finden Sie unter Getting started with Amazon VPC (Erste Schritte mit Amazon VPC) im Amazon-VPC-Benutzerhandbuch.
- Ein Amazon-S3-Endpunkt, der an die VPC angeschlossen ist. Zum Beispiel: vpc-01685961063b0d84b. Weitere Informationen finden Sie unter Endpoints for Amazon S3 (Endpunkte für Amazon S3) im Amazon-VPC-Benutzerhandbuch.



- Ein Routeneintrag, der auf den VPC-Endpunkt verweist. Zum Beispiel vpce-0ec5da4d265227786 in der Routing-Tabelle, die vom VPC-Endpunkt verwendet wird (vpce-0ec5da4d265227786).



- Eine Netzwerk-ACL, die an die VPC angeschlossen ist, erlaubt den Datenverkehr.
- Eine an die VPC angeschlossene Sicherheitsgruppe erlaubt den Datenverkehr.

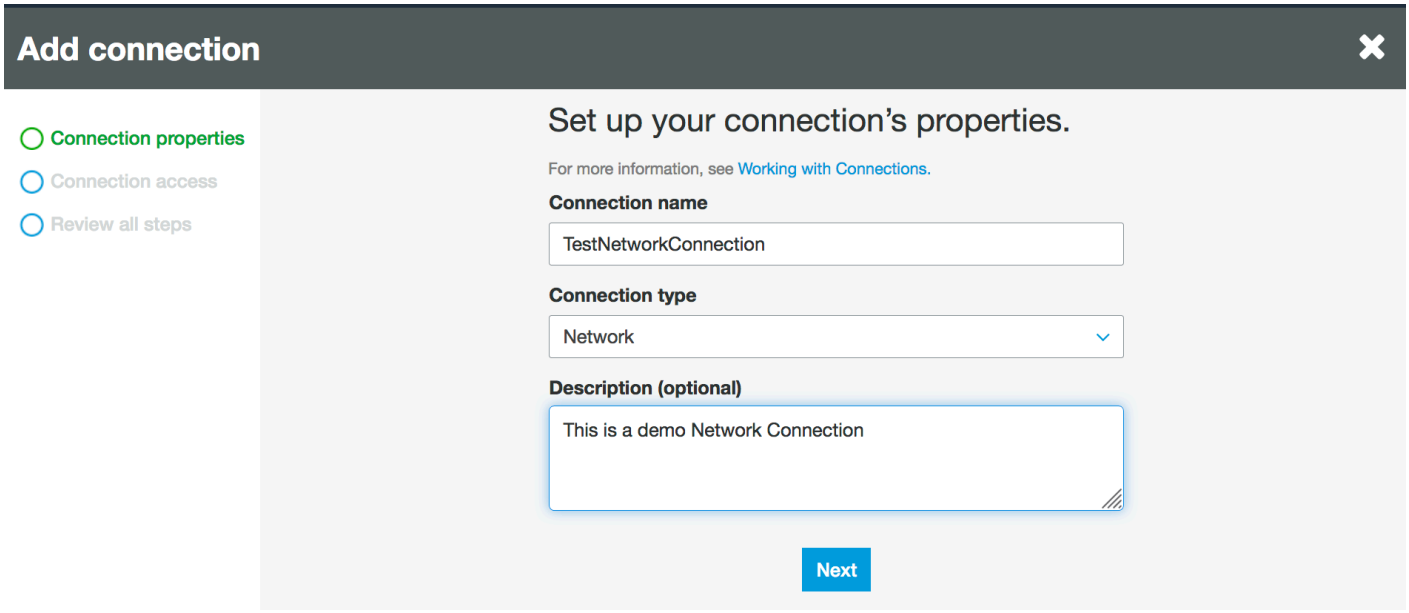
Herstellen der Verbindung zu Amazon S3

Typischerweise erstellen Sie diese Ressourcen innerhalb von Amazon Virtual Private Cloud (Amazon VPC), so dass sie nicht über das öffentliche Internet zugänglich sind. Standardmäßig AWS Glue kann nicht auf Ressourcen innerhalb einer VPC zugegriffen werden. Um den Zugriff auf Ressourcen innerhalb Ihrer VPC AWS Glue zu ermöglichen, müssen Sie zusätzliche VPC-spezifische Konfigurationsinformationen angeben, die das IDs VPC-Subnetz und die Sicherheitsgruppe umfassen. IDs Um eine `Network`-Verbindung herzustellen, benötigen Sie folgende Informationen:

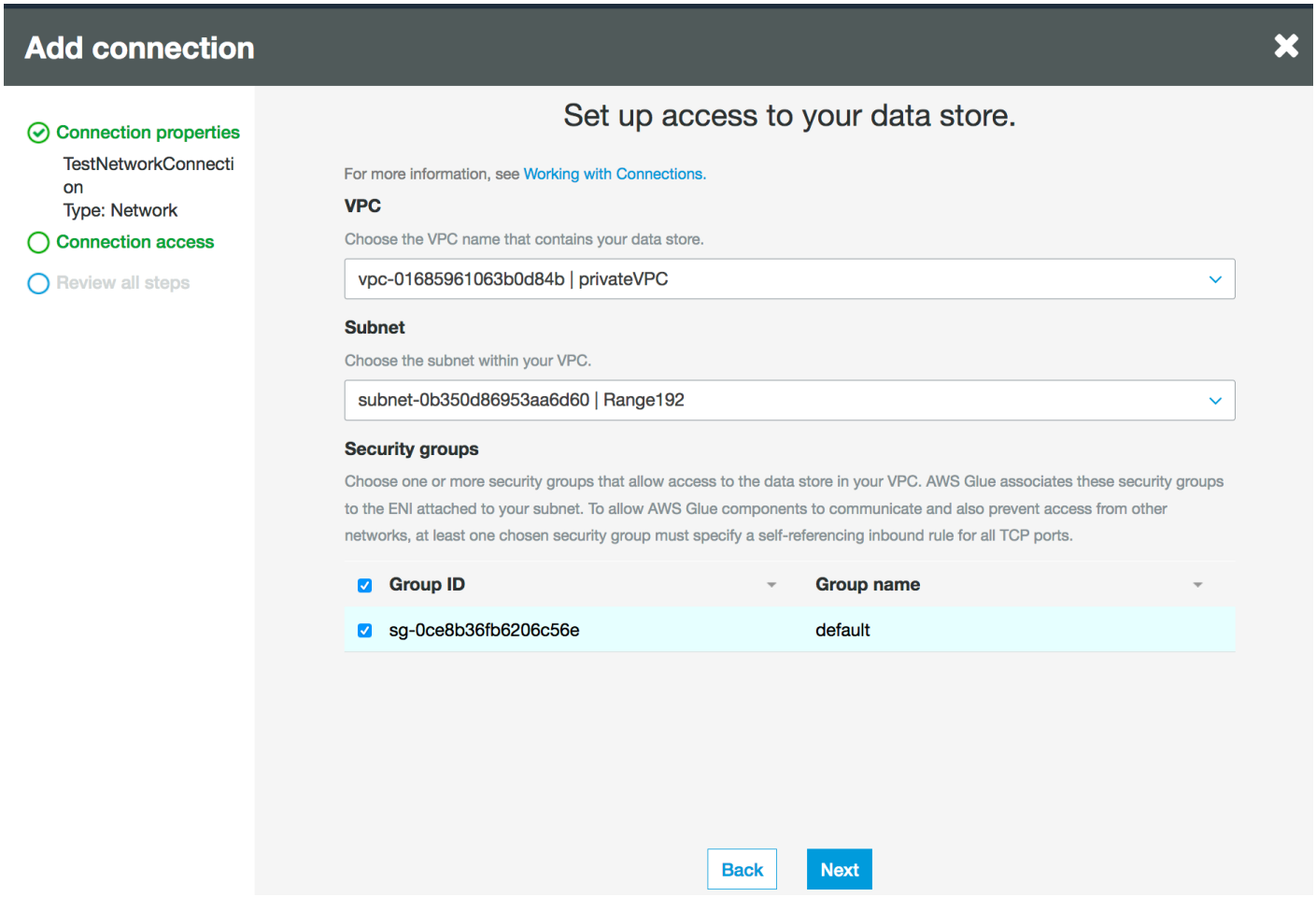
- EINE VPC-ID
- Ein Subnetz innerhalb der VPC
- Eine Sicherheitsgruppen-ID

Eine `Network`-Verbindung einrichten:

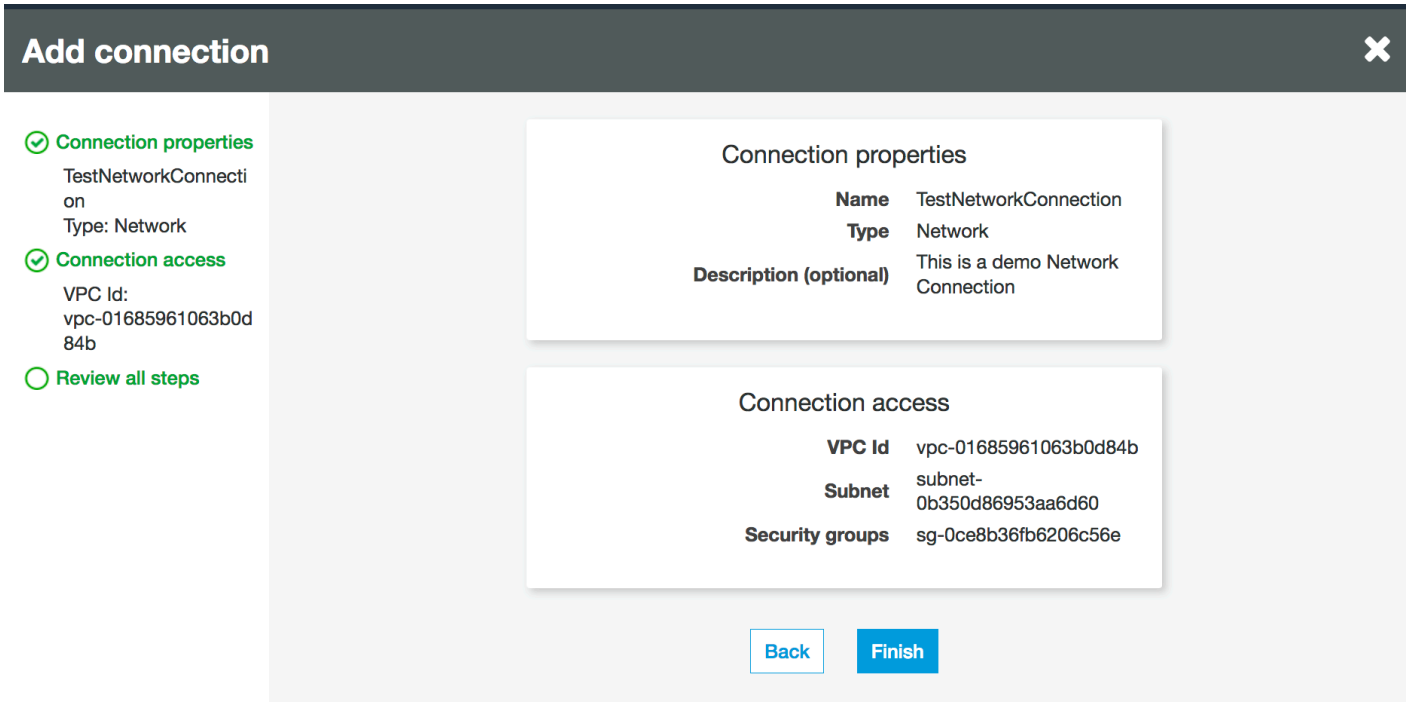
1. Wählen Sie im Navigationsbereich der AWS Glue -Konsole Add connection (Verbindung hinzufügen) aus.
2. Geben Sie den Verbindungsnamen ein und wählen Sie Network (Netzwerk) als Verbindungstyp. Wählen Sie Next (Weiter).



3. Konfigurieren Sie die VPC -, Subnetz- und Sicherheitsgruppeninformationen.
   - VPC: Wählen Sie den Namen der VPC aus, der Ihren Datenspeicher enthält.
   - Subnetz: Wählen Sie das Subnetz in Ihrer VPC aus.
   - Sicherheitsgruppen: Wählen Sie eine oder mehrere Sicherheitsgruppen aus, die den Zugriff auf den Datenspeicher in Ihrer VPC ermöglichen.



4. Wählen Sie Weiter.
5. Überprüfen Sie die Verbindungsinformationen und wählen Sie Finish (Beenden) aus.



Testen der Verbindung zu Amazon S3

Sobald Sie Ihre `Network`-Verbindung erstellt haben, können Sie die Konnektivität zu Ihrem Amazon-S3-Datenspeicher in einem VPC-Endpunkt testen.

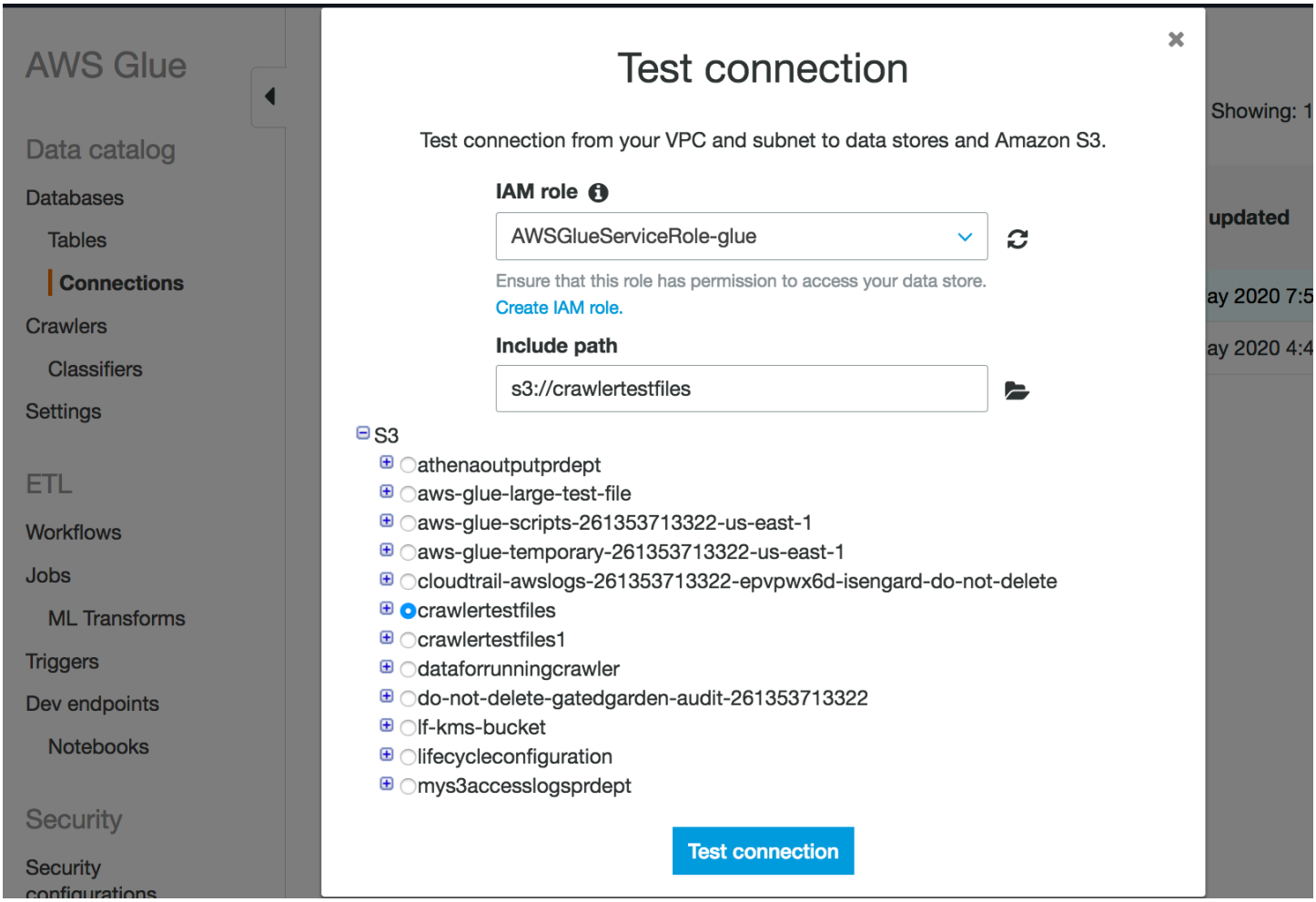
Beim Testen einer Verbindung können folgende Fehler auftreten:

- INTERNET-VERBINDUNGSFEHLER: zeigt ein Problem mit der Internetverbindung an
- UNGÜLTIGER BUCKET FEHLER: deutet auf ein Problem mit dem Amazon S3 Bucket hin
- S3 VERBINDUNGSFEHLER: zeigt einen Fehler bei der Verbindung zu Amazon S3 an
- UNGÜLTIGER VERBINDUNGSTYP: weist darauf hin, dass der Verbindungstyp nicht den erwarteten Wert hat, `NETWORK`
- UNGÜLTIGER VERBINDUNGSTESTTYP: weist auf ein Problem mit dem Typ des Netzwerkverbindungstests hin
- INVALID TARGET: weist darauf hin, dass der Amazon S3 Bucket nicht richtig angegeben wurde

So testen Sie eine `Network`-Verbindung:

1. Wählen Sie die Netzwerkverbindung in der AWS Glue -Konsole aus.
2. Wählen Sie Test connection (Verbindung testen) aus.

3. Wählen Sie die IAM-Rolle aus, die Sie im vorherigen Schritt erstellt haben, und geben Sie einen Amazon S3 Bucket an.
4. Wählen Sie zum Starten des Tests Test connection (Verbindung testen) aus. Es kann einige Augenblicke dauern, um das Ergebnis zu zeigen.



Wenn Ihnen ein Fehler angezeigt wird, gehen Sie folgendermaßen vor:

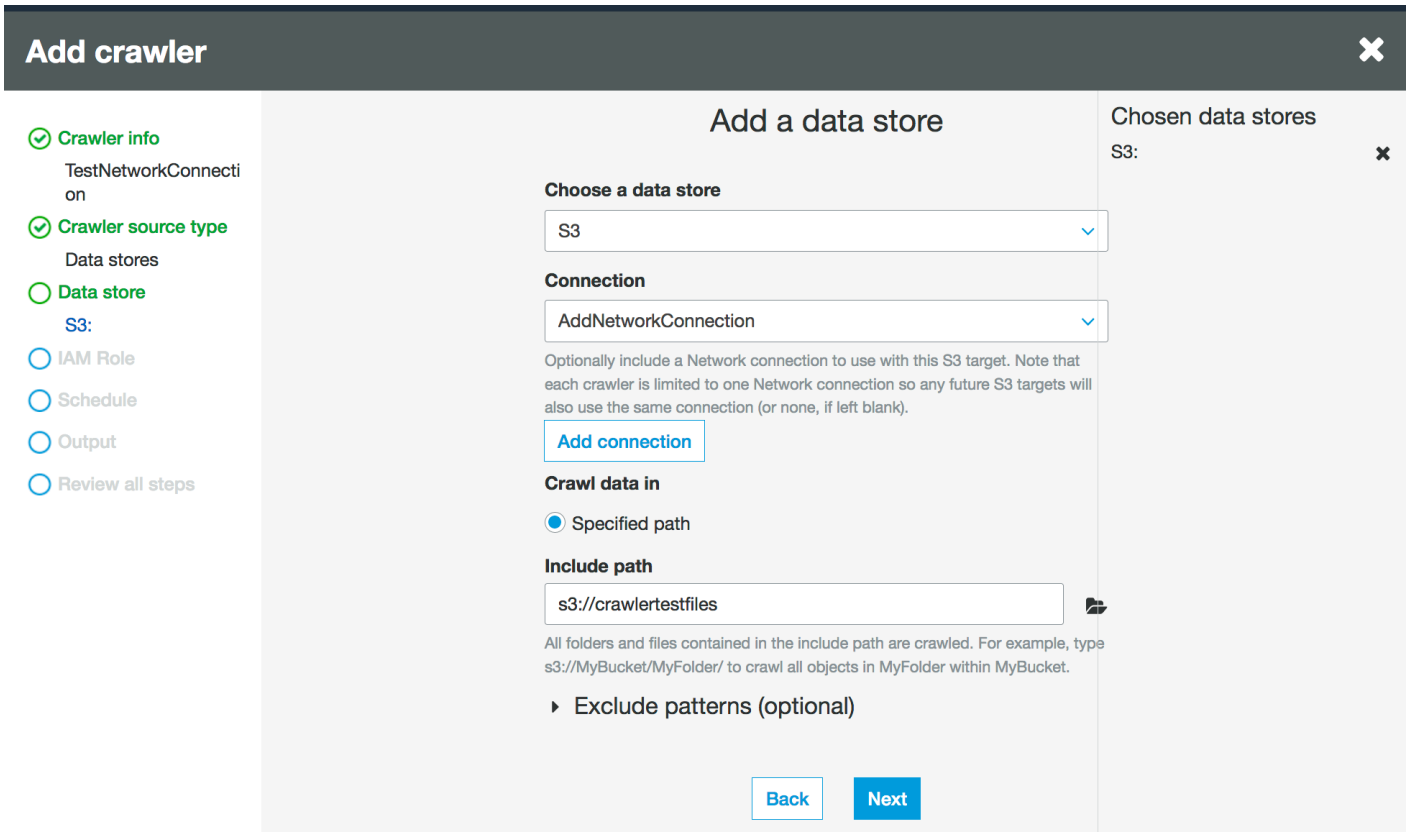
- Die richtigen Berechtigungen werden für die ausgewählte Rolle bereitgestellt.
- Der richtige Amazon S3 Bucket wird bereitgestellt.
- Die Sicherheitsgruppen und die Netzwerk-ACL ermöglichen den erforderlichen eingehenden und ausgehenden Datenverkehr.
- Die angegebene VPC ist mit einem Amazon-S3-VPC-Endpunkt verbunden.

Sobald Sie die Verbindung erfolgreich getestet haben, können Sie einen Crawler erstellen.

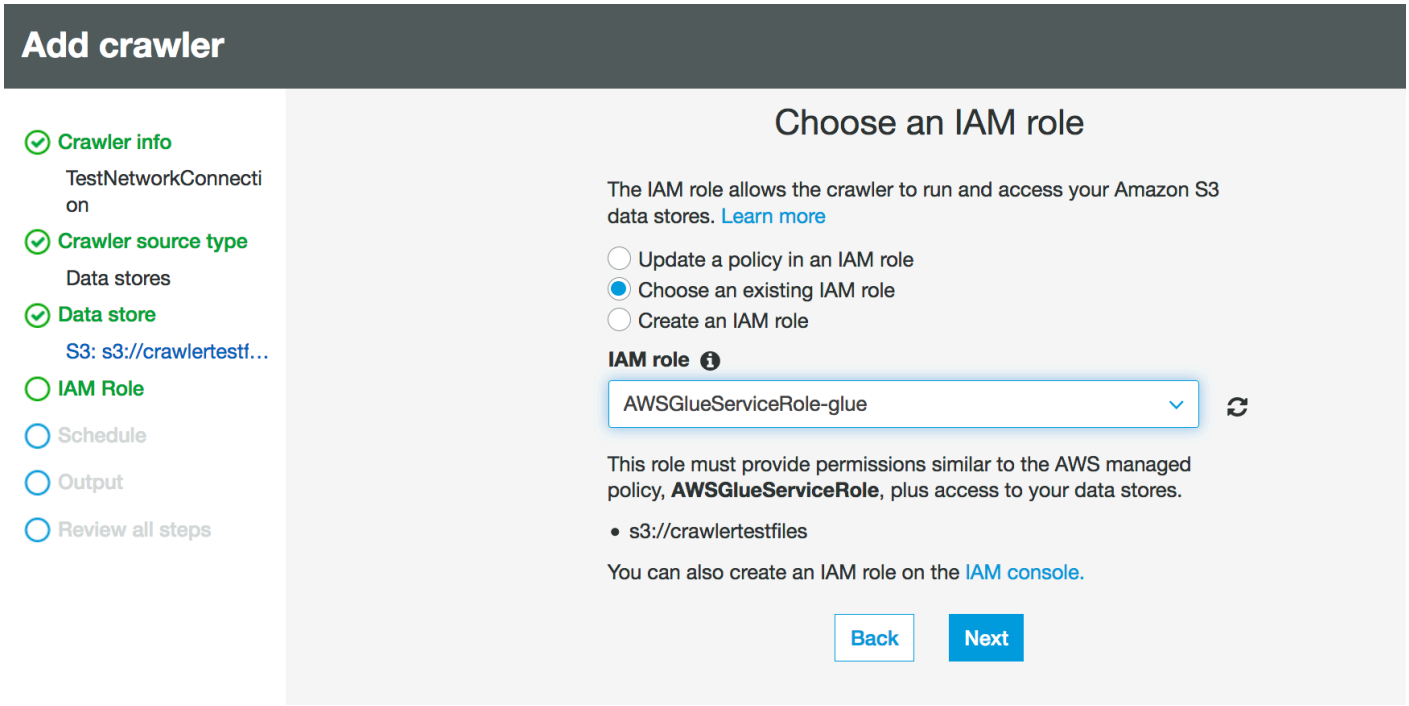
Erstellen eines Crawlers für einen Amazon-S3-Datenspeicher

Jetzt können Sie einen Crawler erstellen, der die `Network`-Verbindung angibt, die Sie erstellt haben. Weitere Informationen zum Erstellen eines Crawlers finden Sie unter Konfiguration eines Crawlers.

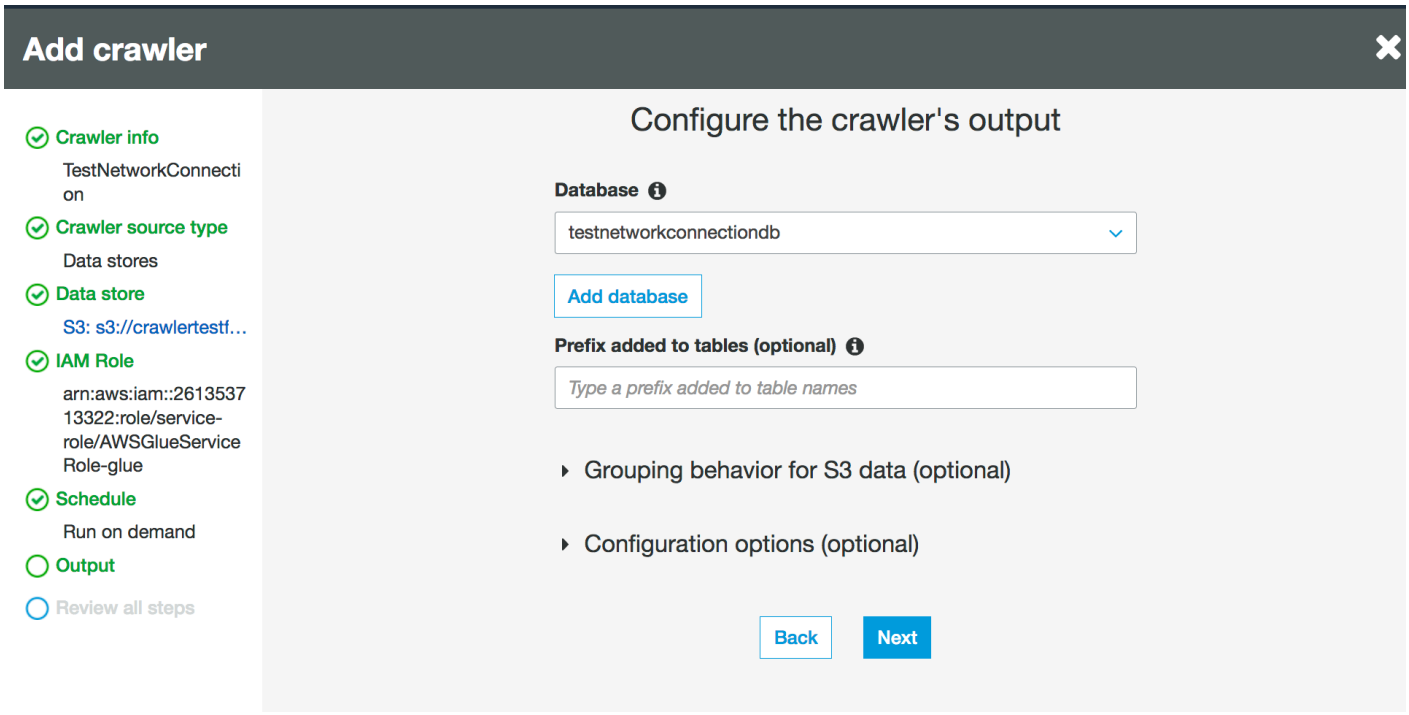
1. Wählen Sie zunächst im Navigationsbereich der Konsole Crawlers aus. AWS Glue
2. Wählen Sie Add crawler (Crawler hinzufügen).
3. Geben Sie dem Crawler einen Namen und klicken Sie auf Next (Weiter).
4. Wählen Sie bei der Aufforderung zur Datenquelle S3 aus und geben Sie das Präfix für den Amazon S3 Bucket und die Verbindung an, die Sie zuvor erstellt haben.



5. Fügen Sie ggf. einen anderen Datenspeicher auf derselben Netzwerkverbindung hinzu.
6. Wählen Sie die IAM-Rolle aus. Die IAM-Rolle muss den Zugriff auf den AWS Glue -Service und den Amazon S3 Bucket gestatten. Weitere Informationen finden Sie unter the section called “Konfiguration eines Crawlers”.



7. Definieren Sie den Zeitplan für den Crawler.
8. Wählen Sie eine vorhandene Datenbank im Data Catalog aus oder erstellen Sie einen neuen Datenbankeintrag.

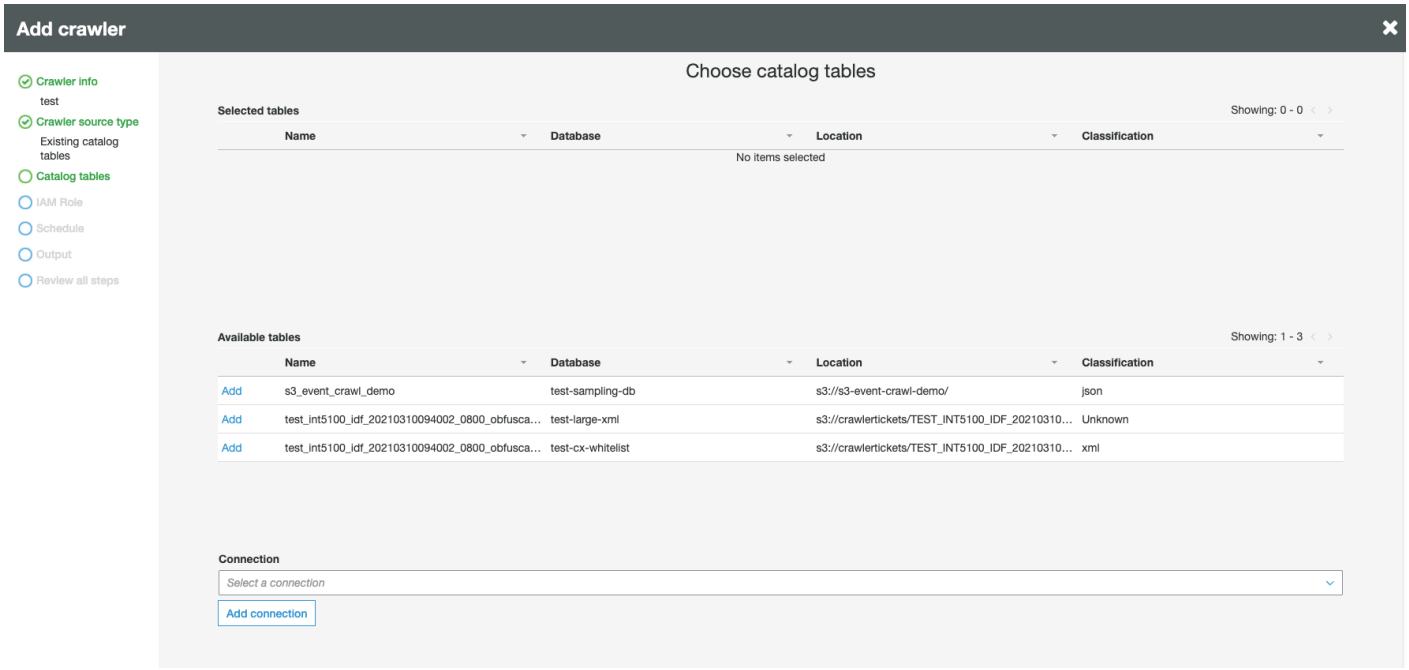


9. Beenden Sie die verbleibende Einrichtung.

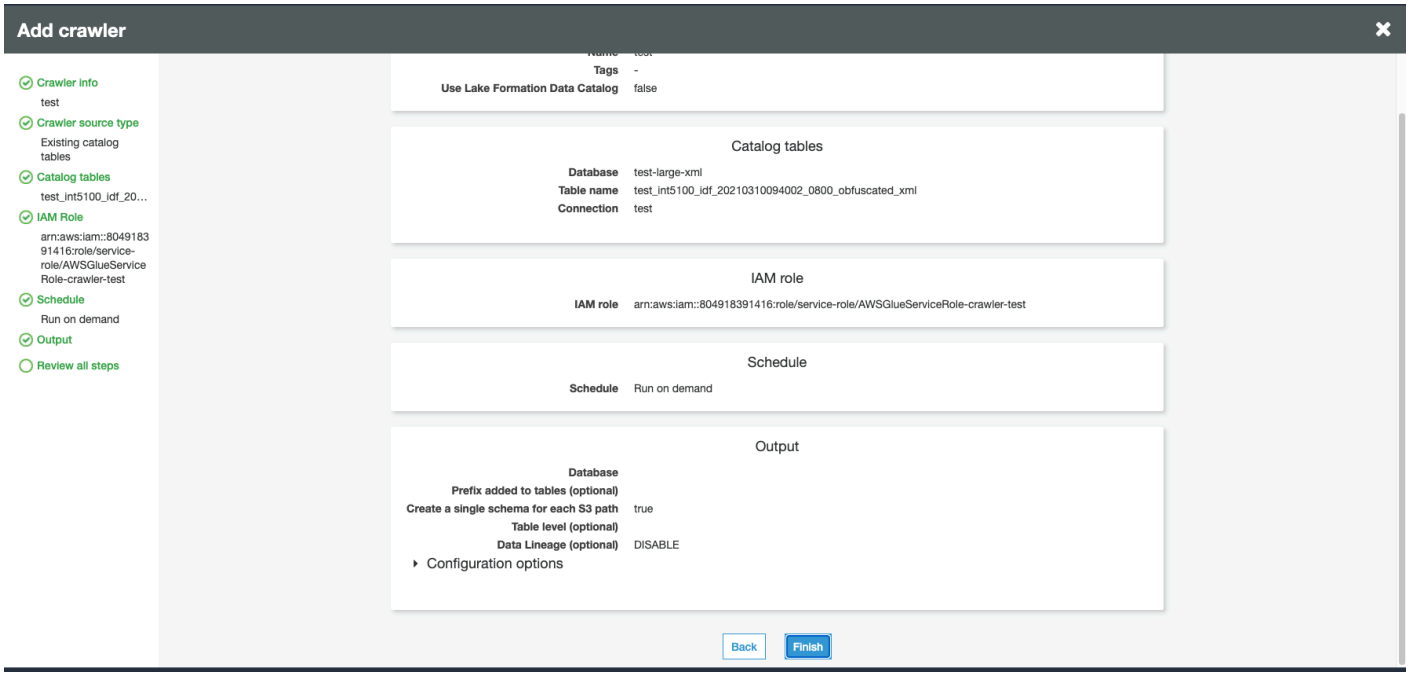
Erstellen eines Crawlers für Amazon-S3-unterstützte Datenkatalog-Tabellen

Sie können jetzt einen Crawler erstellen, der die von Ihnen erstellte `Network`-Verbindung und einen Katalogquelltyp angibt. Weitere Informationen zum Erstellen eines Crawlers finden Sie unter Konfiguration eines Crawlers.

1. Wählen Sie zunächst im Navigationsbereich der Konsole Crawlers aus. AWS Glue
2. Wählen Sie Add crawler (Crawler hinzufügen).
3. Geben Sie dem Crawler einen Namen und klicken Sie auf Next (Weiter).
4. Wenn Sie nach dem Crawler-Quellentyp gefragt werden, wählen Sie Bestehende Katalogtabellen, und geben Sie die vorhandenen Katalogtabellen an, die aus der Liste der verfügbaren Tabellen gecrawlt werden sollen.

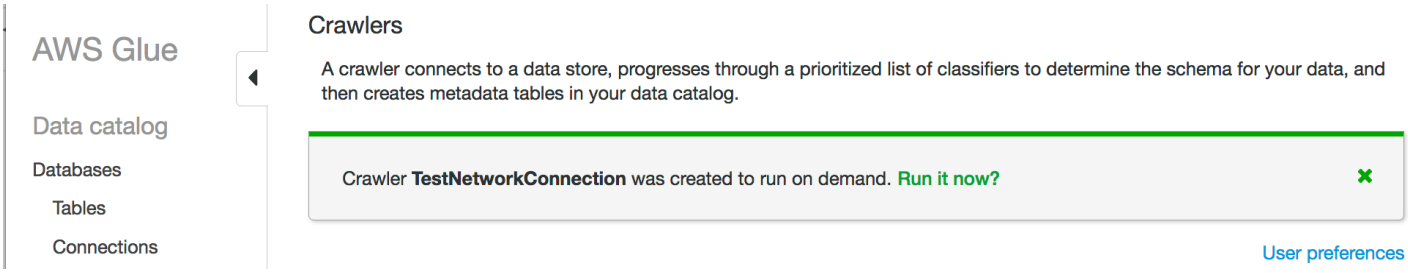


5. Wählen Sie die IAM-Rolle aus. Die IAM-Rolle muss den Zugriff auf den AWS Glue -Service und den Amazon S3 Bucket gestatten. Weitere Informationen finden Sie unter the section called “Konfiguration eines Crawlers”.
6. Definieren Sie den Zeitplan für den Crawler.
7. Wählen Sie eine vorhandene Datenbank im Data Catalog aus oder erstellen Sie einen neuen Datenbankeintrag.
8. Beenden Sie die verbleibende Einrichtung und überprüfen Sie Ihre Schritte.



Ausführen eines Crawlers

Führen Sie Ihren Crawler aus.



Fehlerbehebung

Informationen zur Fehlerbehebung in Bezug auf Amazon S3 Buckets mit einem VPC Gateway finden Sie unter Why can't I connect to an S3 bucket using a gateway VPC endpoint? (Warum kann ich keine Verbindung zu einem S3 Bucket unter Verwendung eines Gateway-VPC-Endpunkts herstellen?)

Arbeiten mit AWS Lake Formation-geschützten Daten

Vollständiger Tabellenzugriff

AWS Glue unterstützt Full Table Access (FTA) für AWS Lake Formation-geschützte Tabellen. Dadurch können Ihre ETL-Jobs Daten mit vollständigen Tabellenberechtigungen lesen und schreiben.

Voraussetzungen

- Geeignete IAM-Rollen und -Berechtigungen
- AWS Lake Formation für Ihren Datenkatalog konfiguriert
- Kompatible Tabellentypen (Hive oder Iceberg)

Wichtigste Überlegungen

Erforderliche Berechtigungen

- `lakeformation:GetDataAccess`IAM-Genehmigung
- AWS Lake Formation Tabellenberechtigungen
- Zugriffsberechtigungen für Amazon S3 S3-Buckets

Unterstützte Tabellentypen

- Hive-Tabellen
- Iceberg-Tabellen

Einschränkungen

- Nicht kompatibel mit Spark Streaming
- Kann nicht gleichzeitig mit einer detaillierten Zugriffskontrolle verwendet werden
- Unterstützt keine Delta- oder Hudi-Tabellen

Bewährte Methoden

1. Stellen Sie vor der Ausführung des Jobs sicher, dass die IAM-Rollen ordnungsgemäß konfiguriert sind
2. Testen Sie die Zugriffsberechtigungen in einer Entwicklungsumgebung
3. Überwachen Sie die Protokolle zur Auftragsausführung auf Probleme im Zusammenhang mit Berechtigungen
4. Sorgen Sie für eine klare Dokumentation der Zugriffsmuster

Fehlerbehebung

Zu den häufigsten Problemen gehören:

- Fehlende IAM-Berechtigungen
- Falsche Konfiguration AWS Lake Formation
- Kompatibilitätsprobleme mit dem Tabellentyp

Vollständige Anweisungen zur Einrichtung und Konfigurationsdetails finden Sie unter Verwendung AWS Lake Formation mit vollständigem Tabellenzugriff.

Behebung von Verbindungsproblemen in AWS Glue

Wenn ein AWS Glue Crawler oder ein Job Verbindungseigenschaften verwendet, um auf einen Datenspeicher zuzugreifen, können Fehler auftreten, wenn Sie versuchen, eine Verbindung herzustellen. AWS Glue verwendet private IP-Adressen im Subnetz, wenn elastische Netzwerkschnittstellen in Ihrer angegebenen Virtual Private Cloud (VPC) und dem Subnetz erstellt werden. In der -Verbindung angegebene Sicherheitsgruppen werden auf die einzelnen Elastic Network-Schnittstellen angewendet. Überprüfen Sie, ob Sicherheitsgruppen den ausgehenden Zugriff zulassen und ob sie Verbindungen zum Datenbank-Cluster zulassen.

Darüber hinaus erfordert Apache Spark bidirektionale Verbindungen zwischen Treiber und Executor-Knoten. Eine der Sicherheitsgruppen muss Regeln für eingehenden Datenverkehr auf allen TCP-Ports zulassen. Sie können verhindern, dass sie der Welt offen steht, indem Sie die Quelle der Sicherheitsgruppe mit einer selbstreferenzierenden Sicherheitsgruppe auf sich selbst beschränken.

Hier finden Sie einige typische Maßnahmen, die Sie ergreifen können, um Verbindungsprobleme zu beheben:

- Prüfen Sie die Port-Adresse Ihrer Verbindung.
- Überprüfen Sie den Benutzernamen und die Passwort-Zeichenfolge in Ihrer Verbindung oder Ihrem Secret.
- Überprüfen Sie bei einem JDBC-Datenspeicher, ob dieser eingehende Verbindungen zulässt.
- Überprüfen Sie, ob innerhalb Ihrer VPC auf Ihren Datenspeicher zugegriffen werden kann.
- Wenn Sie Ihre Verbindungsdaten mit speichern, stellen Sie sicher AWS Secrets Manager, dass Ihre IAM-Rolle für die Zugriffsberechtigung auf Ihr AWS Glue Geheimnis berechtigt ist. Weitere Beispiele finden Sie unter Beispiel: Berechtigung zum Abrufen von Secret-Werten im AWS Secrets Manager -Benutzerhandbuch. Abhängig von Ihrer Netzwerkkonfiguration müssen Sie möglicherweise auch einen VPC-Endpunkt erstellen, um eine private Verbindung zwischen Ihrer VPC und dem Secrets Manager herzustellen. Weitere Informationen finden Sie unter Verwenden eines AWS Secrets Manager VPC-Endpunkts.

Tutorial: Den AWS Glue Connector für Elasticsearch verwenden

Elasticsearch ist ein häufig verwendetes Open-Source-Such- und Analysemodul für die Protokollanalyse, Echtzeitüberwachung von Anwendungen, Clickstream-Analyse und ähnliche Anwendungsfälle. Sie können ihn OpenSearch als Datenspeicher für Ihre ETL-Jobs (Extrahieren, Transformieren und Laden) verwenden, indem Sie den AWS Glue Connector für Elasticsearch in konfigurieren. AWS Glue Studio Dieser Konnektor kann kostenlos von AWS Marketplace bezogen werden.

> **Note**
>
> Der AWS Marketplace Elasticsearch Spark Connector ist veraltet. Bitte verwenden Sie stattdessen den AWS Glue -Konnektor für Elasticsearch .

In diesem Tutorial zeigen wir, wie Sie mit einer minimalen Anzahl von Schritten eine Verbindung zu Ihren Amazon OpenSearch Service-Knoten herstellen können.

Themen

- Voraussetzungen
- Schritt 1: (Optional) Erstellen Sie einen AWS geheimen Schlüssel für Ihre OpenSearch Clusterinformationen
- Schritt 2: Abonnieren des Konnektors

- Schritt 3: Aktivieren Sie den Connector AWS Glue Studio und stellen Sie eine Verbindung her
- Schritt 4: Konfigurieren einer IAM-Rolle für Ihren ETL-Auftrag
- Schritt 5: Erstellen Sie einen Job, der die OpenSearch Verbindung verwendet
- Schritt 6: Ausführen des Auftrags

Voraussetzungen

Für dieses Tutorial benötigen Sie Folgendes:

- Zugriff auf AWS Glue Studio
- Zugriff auf einen OpenSearch Cluster in der AWS Cloud
- (Optional) Zugriff auf AWS Secrets Manager.

Schritt 1: (Optional) Erstellen Sie einen AWS geheimen Schlüssel für Ihre OpenSearch Clusterinformationen

Damit die Anmeldeinformationen für die Verbindung sicher gespeichert sind und Sie sie sicher verwenden können, speichern Sie sie in AWS Secrets Manager. Das Secret, das Sie erstellen, wird später im Tutorial von der Verbindung verwendet. Die Schlüssel-Wert-Paare der Anmeldeinformationen werden als normale Verbindungsoptionen in den AWS Glue Connector für Elasticsearch eingespeist.

Weitere Informationen zur Erstellung von Secrets finden Sie unter Erstellen und Verwalten von Secrets mit AWS Secrets Manager im AWS Secrets Manager -Benutzerhandbuch.

Um ein Geheimnis zu erstellen AWS

1. Melden Sie sich an der AWS Secrets Manager -Konsole an.
2. Wählen Sie entweder auf der Service-Einführungsseite oder der Secrets-Listenseite die Option Store a new secret (Neues Secret speichern).
3. Wählen Sie auf der Seite Store a new secret (Ein neues Secret speichern) die Option Other type of secret (Andere Art von Secret). Bei dieser Option müssen die Struktur und die Details Ihres Secrets angeben.
4. Fügen Sie ein Schlüssel - und Wertepaar für den OpenSearch Cluster-Benutzernamen hinzu. Zum Beispiel:

`es.net.http.auth.user`: *`username`*

5. Wählen Sie + Add row (Zeile hinzufügen) aus und geben Sie ein weiteres Schlüssel-Wert-Paar für das Passwort ein. Zum Beispiel:

   `es.net.http.auth.pass`: *`password`*

6. Wählen Sie Weiter.
7. Geben Sie einen Secret-Namen ein. Beispiel: my-es-secret. Optional können Sie eine Beschreibung eingeben.

   Notieren Sie den Secret-Namen, der später in diesem Tutorial verwendet wird, und wählen Sie Next (Weiter) aus.
8. Wählen Sie Next (Weiter) aus und danach Store (Speichern), um das Secret zu erstellen.

Nächster Schritt

Schritt 2: Abonnieren des Konnektors

Schritt 2: Abonnieren des Konnektors

Der AWS Glue Connector für Elasticsearch ist kostenlos erhältlich bei. AWS Marketplace

Um den AWS Glue Connector für Elasticsearch zu abonnieren unter AWS Marketplace

1. Wenn Sie Ihr AWS Konto noch nicht für die Verwendung von License Manager konfiguriert haben, gehen Sie wie folgt vor:

   a. Öffnen Sie die AWS License Manager Konsole unter https://console.aws.amazon.com/license-manager.

   b. Wählen Sie Create customer managed license (Kundenverwaltete Lizenz erstellen) aus.

   c. Wählen Sie im Fenster IAM-Berechtigungen (einmalige Einrichtung) die Option Ich erteile AWS License Manager die erforderlichen Berechtigungen und anschließend Berechtigungen gewähren aus.

      Wenn Sie dieses Fenster nicht sehen, haben Sie bereits die erforderlichen Berechtigungen konfiguriert.
2. Öffnen Sie AWS Glue Studio Konsole unter. https://console.aws.amazon.com/gluestudio/

3. Erweitern Sie in der AWS Glue Studio Konsole das Menüsymbol
   (☰)
   und wählen Sie dann im Navigationsbereich Connectors aus.
4. Wählen Sie auf der Seite Connectors die Option Go to AWS Marketplace (Zu MKT wechseln) aus.
5. Geben Sie im Bereich AWS Glue Studio Produkte suchen den Text AWS Glue Connector for Elasticsearch in das Suchfeld ein und drücken Sie dann die Eingabetaste. AWS Marketplace
6. Wählen Sie den Namen des Connectors. AWS Glue -Connector für Elasticsearch aus.
7. Die Registerkarten auf der Produktseite für den Konnektor enthalten Informationen zum Konnektor. Wenn Sie bereit sind, fortzufahren, wählen Sie Continue to Subscribe (Abonnieren fortsetzen) aus.
8. Lesen Sie die Nutzungsbedingungen für die Software. Klicken Sie auf Accept Terms(Bedingungen akzeptieren).
9. Wenn der Abonnementvorgang abgeschlossen ist, wird die Benachrichtigung „Vielen Dank, dass Sie dieses Produkt abonniert haben! Sie können jetzt Ihre Software konfigurieren“ angezeigt. Oberhalb des Banners befindet sich die Schaltfläche Continue to Configuration (Weiter zur Konfiguration). Wählen Sie Continue to Configuration (Weiter zur Konfiguration) aus.
10. Wählen Sie die Option „Fulfillment“ auf der Seite Configure this software (Diese Software konfigurieren). Sie können entweder zwischen AWS Glue 1.0/2.0 oder 3.0 wählen. AWS Glue Wählen Sie dann Continue to Launch (Weiter zum Start) aus.

Nächster Schritt

Schritt 3: Aktivieren Sie den Connector AWS Glue Studio und stellen Sie eine Verbindung her

Schritt 3: Aktivieren Sie den Connector AWS Glue Studio und stellen Sie eine Verbindung her

Nachdem Sie Weiter zum Start ausgewählt haben, wird die Seite Diese Software starten unter angezeigt AWS Marketplace. Nachdem Sie den Link zur Aktivierung des Connectors verwendet haben AWS Glue Studio, stellen Sie eine Verbindung her.

Um den Connector bereitzustellen und eine Verbindung herzustellen in AWS Glue Studio

1. Wählen Sie auf der Seite Diese Software starten in der AWS Marketplace Konsole die Option Nutzungsanweisungen und dann den Link im daraufhin angezeigten Fenster aus.

Ihr Browser wird auf die AWS Glue Studio Konsolenseite Marketplace-Verbindung erstellen umgeleitet.

2. Geben Sie einen Namen für die Verbindung ein. Beispiel: my-es-connection.
3. Wählen Sie im Abschnitt Connection access (Verbindungszugriff) bei Connection credential type (Anmeldeinformationstyp für die Verbindung) die Option Benutzername und Passwort aus.
4. Geben Sie als AWS secret den Namen Ihres Secrets ein. Beispiel: my-es-secret.
5. Geben Sie im Abschnitt Netzwerkoptionen die VPC-Informationen für die Verbindung mit dem OpenSearch Cluster ein.
6. Wählen Sie Create connection and activate connector (Verbindung erstellen und Konnektor aktivieren) aus.

Nächster Schritt

Schritt 4: Konfigurieren einer IAM-Rolle für Ihren ETL-Auftrag

Schritt 4: Konfigurieren einer IAM-Rolle für Ihren ETL-Auftrag

Wenn Sie den AWS Glue ETL-Job erstellen, geben Sie eine AWS Identity and Access Management (IAM-) Rolle an, die der Job verwenden soll. Die Rolle muss Zugriff auf alle vom Job verwendeten Ressourcen gewähren, einschließlich Amazon S3 (für alle Quellen, Ziele, Skripts, Treiberdateien und temporären Verzeichnisse) und auch auf AWS Glue Data Catalog Objekte.

Die angenommene IAM-Rolle für den AWS Glue ETL-Job muss auch Zugriff auf das Geheimnis haben, das im vorherigen Abschnitt erstellt wurde. Standardmäßig hat die verwaltete AWS-Rolle `AWSGlueServiceRole` keinen Zugriff auf das Secret. Informationen zum Einrichten der Zugriffssteuerung für Ihre Secrets finden Sie unter Authentifizierung und Zugriffskontrolle für AWS Secrets Manager und Beschränken des Zugriffs auf bestimmte Secrets.

Eine IAM-Rolle für den ETL-Auftrag konfigurieren

1. Konfigurieren Sie die in the section called “Erforderliche IAM-Berechtigungen für ETL-Aufträge überprüfen” beschriebenen Berechtigungen.
2. Konfigurieren Sie die zusätzlichen Berechtigungen, die für die Verwendung von Connectoren mit erforderlich AWS Glue Studio sind, wie unter beschriebenthe section called “Erforderliche Berechtigungen zur Verwendung von Konnektoren”.

Nächster Schritt

Schritt 5: Erstellen Sie einen Job, der die OpenSearch Verbindung verwendet

Schritt 5: Erstellen Sie einen Job, der die OpenSearch Verbindung verwendet

Nachdem Sie eine Rolle für Ihren ETL-Job erstellt haben, können Sie einen Job erstellen AWS Glue Studio , der die Verbindung und den Konnektor für Open Spark verwendet ElasticSearch.

Wenn Ihr Auftrag in einer Amazon Virtual Private Cloud (Amazon VPC) ausgeführt wird, stellen Sie sicher, dass die VPC korrekt konfiguriert ist. Weitere Informationen finden Sie unter the section called “Konfigurieren Sie eine VPC für Ihren ETL-Auftrag”.

Einen Auftrag erstellen, der den Elasticsearch-Spark-Konnektor verwendet

1. Wählen Sie in AWS Glue StudioConnectors aus.
2. Wählen Sie aus der Liste Your connection (Ihre Verbindung) die soeben erstellte Verbindung und dann Create job (Auftrag erstellen) aus.
3. Wählen Sie im visuellen Auftragseditor den Datenquellknoten aus. Auf der Registerkarte rechts (Data source properties - Connector (Datenquelleigenschaften – Konnektor)) machen Sie zusätzliche Angaben für den Konnektor.

    a. Wählen Sie Add schema (Schema hinzufügen) und geben Sie das Schema des Datensatzes in der Datenquelle ein. Verbindungen verwenden keine Tabellen, die im Datenkatalog gespeichert sind, was bedeutet, dass AWS Glue Studio der Datenkatalog das Schema der Daten nicht kennt. Sie müssen diese Schemainformationen manuell angeben. Anweisungen zur Verwendung des Schema-Editors finden Sie unter the section called “Bearbeiten des Schemas in einem benutzerdefinierten Transformationsknoten”.

    b. Klappen Sie Connection options (Verbindungsoptionen) aus.

    c. Wählen Sie die Option Neue hinzufügen und geben Sie die Informationen ein, die für den Konnektor benötigt werden, der nicht in das AWS Geheimnis eingegeben wurde:

    - es.nodes: https://< OpenSearch Domain-Endpunkt>
    - es.port: 443
    - Pfad: Test
    - es.nodes.wan.only.: true

Eine Erläuterung dieser Verbindungsoptionen finden Sie unter https://www.elastic.co/guide/en/elasticsearch/hadoop/current/configuration.html.

4. Fügen Sie dem Diagramm einen Zielknoten hinzu.

   Ihr Datenziel kann Amazon S3 sein, es kann aber auch Informationen aus einem AWS Glue Data Catalog oder einen Connector nutzen, um Daten an einem anderen Speicherort zu schreiben. Sie können beispielsweise mithilfe einer Data-Catalog-Tabelle in eine Datenbank in Amazon RDS schreiben oder einen Connector als Datenziel verwenden, um in Datenspeicher zu schreiben, die von sich aus nicht in AWS Glue unterstützt werden.

   Wenn Sie einen Connector für Ihr Datenziel auswählen, müssen Sie eine Verbindung auswählen, die für diesen Connector erstellt wurde. Außerdem müssen Sie, falls der Connectoranbieter das verlangt, Optionen hinzufügen, um dem Connector zusätzliche Informationen bereitzustellen. Wenn Sie eine Verbindung verwenden, die Informationen für ein AWS Geheimnis enthält, müssen Sie den Benutzernamen und das Kennwort nicht in den Verbindungsoptionen angeben.

5. Fügen Sie optional zusätzliche Datenquellen und einen oder mehrere Transformationsknoten hinzu, wie unter the section called “Transformieren Sie Daten mit AWS Glue verwaltete Transformationen” beschrieben.

6. Konfigurieren Sie die Auftragseigenschaften wie unter the section called “Ändern der Auftragseigenschaften” beschrieben, beginnend mit Schritt 3, und speichern Sie den Auftrag.

Nächster Schritt

Schritt 6: Ausführen des Auftrags

Schritt 6: Ausführen des Auftrags

Nachdem Sie den Auftrag gespeichert haben, können Sie ihn ausführen. Dadurch starten die ETL-Operationen.

Um den Job auszuführen, den Sie für den AWS Glue Connector for Elasticsearch erstellt haben

1. Wählen Sie in der AWS Glue Studio Konsole auf der Visual Editor-Seite die Option Ausführen aus.

2. Um Informationen über die Auftragsausführung zu sehen, wählen Sie die Option Run Details (Details ausführen) im Bestätigungsbanner oder die Registerkarte Runs (Ausführungen) im visuellen Editor.

AWS Glue Jobs mit interaktiven Sessions aufbauen

Dateningenieure können Inhalte verfassen AWS Glue Jobs schneller und einfacher als zuvor mithilfe interaktiver Sitzungen in AWS Glue.

Themen

- Überblick über AWS Glue interaktive Sessions
- Erste Schritte mit AWS Glue interaktive Sessions
- Konfigurieren von AWS Glue-interaktiven Sitzungen für Jupyter und AWS Glue Studio-Notebooks
- Ein Skript oder ein Notizbuch in einen AWS Glue Job umwandeln
- Arbeiten mit Streaming-Vorgängen in AWS Glue interaktive Sitzungen
- Lokales Entwickeln und Testen von AWS Glue-Jobskripten
- Entwicklungsendpunkte

Überblick über AWS Glue interaktive Sessions

Mit AWS Glue Mit interaktiven Sitzungen können Sie schnell Datenvorbereitungs- und Analyseanwendungen erstellen, testen und ausführen. Interaktive Sitzungen bieten eine programmatische und visuelle Schnittstelle zum Erstellen und Testen von Extraktions-, Transformations- und Ladeskripten (ETL) für die Datenaufbereitung. Interaktive Sitzungen führen Apache Spark-Analyseanwendungen aus und bieten On-Demand-Zugriff auf eine externe Spark-Laufzeitumgebung. AWS Glue verwaltet transparent serverloses Spark für diese interaktiven Sitzungen.

Interaktive Sitzungen sind flexibel, sodass Sie Ihre Anwendungen in der Umgebung Ihrer Wahl erstellen und testen können. Sie können interaktive Sitzungen über die und die API erstellen AWS Command Line Interface und mit ihnen arbeiten. Sie können Jupyter-kompatible Notebooks verwenden, um Ihre Notebook-Skripte visuell zu erstellen und zu testen. Interaktive Sitzungen bieten einen Open-Source-Jupyter-Kernel, der fast überall integriert werden kann, was Jupyter tut, einschließlich der Integration mit IntelliJ und IDEs VS Code. PyCharm Auf diese Weise können Sie Code in Ihrer lokalen Umgebung erstellen und im Backend der interaktiven Sitzungen nahtlos ausführen.

Mithilfe der Interactive-Sessions-API können Kunden programmgesteuert Anwendungen ausführen, die Apache-Spark-Analysen verwenden, ohne die Spark-Infrastruktur verwalten zu müssen. Sie können eine oder mehrere Spark-Anweisungen in einer einzigen interaktiven Sitzung ausführen.

Interaktive Sitzungen sind daher eine schnellere, günstigere und flexiblere Möglichkeit, Datenvorbereitungs- und Analyseanwendungen zu erstellen und auszuführen. Weitere Informationen zur Verwendung interaktiver Sitzungen finden Sie in der Dokumentation in diesem Abschnitt. Magics wird unterstützt von AWS Glue

Einschränkungen

- Auftrags-Lesezeichen werden in interaktiven Sitzungen nicht unterstützt.
- Das Erstellen von Notebook-Jobs mit dem AWS Command Line Interface wird nicht unterstützt.
- AWS Glue Studio Notebooks unterstützen Scala nicht.

Erste Schritte mit AWS Glue interaktive Sessions

In diesen Abschnitten wird die Ausführung beschrieben AWS Glue interaktive Sitzungen lokal.

Voraussetzungen für die lokale Einrichtung von interaktiven Sitzungen

Für die Installation interaktiver Sitzungen gelten folgende Voraussetzungen:

- Unterstützte Python-Versionen sind 3.6 – 3.10+.
- Anweisungen für MacOS/Linux und Windows finden Sie in den folgenden Abschnitten.

Installation von Jupyter und AWS Glue interaktiven Sitzungen (Jupyter-Kernel)

Verwenden Sie Folgendes, um den Kernel lokal zu installieren.

Der Befehl `install-glue-kernels` installiert die Jupyter-Kernelspezifikation sowohl für den Pyspark- als auch den Spark-Kernel und installiert außerdem Logos im richtigen Verzeichnis.

```
pip3 install --upgrade jupyter boto3 aws-glue-sessions
```

```
install-glue-kernels
```

Ausführen von Jupyter

Führen Sie die folgenden Schritte aus, um Jupyter Notebook auszuführen.

1. Führen Sie zum Starten von Jupyter Notebook den folgenden Befehl aus.

```
jupyter notebook
```

2. Wählen Sie Neu und wählen Sie dann eine der AWS Glue Kernel, gegen die mit dem Programmieren begonnen werden soll AWS Glue.

Konfigurieren der Anmeldeinformationen und der Region für die Sitzung

MacOS/Linux-Anweisungen

AWS Glue interaktive Sitzungen erfordern dieselben IAM-Berechtigungen wie AWS Glue Jobs und Entwicklungsendpunkte. Sie können die Rolle, die mit interaktiven Sitzungen verwendet wird, auf zwei Arten angeben:

1. Mit der `%iam_role` und mit `%region`-Magics
2. Mit einer zusätzlichen Zeile in `~/.aws/config`

Configuring a session role with magic (Konfigurieren einer Sitzungsrolle mit Magic)

Geben Sie in der ersten ausgeführten Zelle den Wert `%iam_role <YourGlueServiceRole>` ein.

Konfigurieren einer Sitzungsrolle mit **`~/.aws/config`**

AWS Glue Die Servicerolle für interaktive Sitzungen kann entweder im Notizbuch selbst angegeben oder zusammen mit der AWS CLI Konfiguration gespeichert werden. Wenn Sie eine Rolle haben, die Sie normalerweise mit verwenden AWS Glue Jobs, das wird diese Rolle sein. Wenn Sie keine Rolle haben, für die Sie sie verwenden AWS Glue Jobs, folgen Sie bitte dieser Anleitung, Konfiguration von IAM-Berechtigungen für AWS Glue, um einen einzurichten.

So legen Sie diese Rolle als Standardrolle für interaktive Sitzungen fest:

1. Öffnen Sie `~/.aws/config` in einem Texteditor.
2. Suchen Sie nach dem Profil, für das Sie es verwenden AWS Glue. Wenn Sie kein Profil verwenden, verwenden Sie das `[Default]` Profil.

3. Fügen Sie im Profil eine Zeile für die Rolle hinzu, die Sie verwenden möchten, z. B. `glue_role_arn=<AWSGlueServiceRole>`.
4. [Optional]: Wenn für Ihr Profil keine Standardregion festgelegt ist, sollten Sie eine mit `region=us-east-1` hinzufügen und `us-east-1` durch die gewünschte Region ersetzen.
5. Speichern Sie die Konfiguration.

Weitere Informationen finden Sie unter Interaktive Sitzungen mit IAM.

Anweisungen für Windows

AWS Glue Für interaktive Sitzungen sind dieselben IAM-Berechtigungen erforderlich wie AWS Glue Jobs und Entwicklungsendpunkte. Sie können die Rolle, die mit interaktiven Sitzungen verwendet wird, auf zwei Arten angeben:

1. Mit der `%iam_role` und mit `%region`-Magics
2. Mit einer zusätzlichen Zeile in `~/.aws/config`

Configuring a session role with magic (Konfigurieren einer Sitzungsrolle mit Magic)

Geben Sie in der ersten ausgeführten Zelle den Wert `%iam_role <YourGlueServiceRole>` ein.

Konfigurieren einer Sitzungsrolle mit **`~/.aws/config`**

AWS Glue Die Servicerolle für interaktive Sitzungen kann entweder im Notizbuch selbst angegeben oder zusammen mit der AWS CLI Konfiguration gespeichert werden. Wenn Sie eine Rolle haben, die Sie normalerweise mit verwenden AWS Glue Jobs, das wird diese Rolle sein. Wenn Sie keine Rolle haben, für die Sie sie verwenden AWS Glue Jobs, folgen Sie bitte dieser Anleitung, Einrichtung von IAM-Berechtigungen für AWS Glue, um einen einzurichten.

So legen Sie diese Rolle als Standardrolle für interaktive Sitzungen fest:

1. Öffnen Sie `~/.aws/config` in einem Texteditor.
2. Suchen Sie nach dem Profil, für das Sie es verwenden AWS Glue. Wenn Sie kein Profil verwenden, verwenden Sie das `[Default]` Profil.
3. Fügen Sie im Profil eine Zeile für die Rolle hinzu, die Sie verwenden möchten, z. B. `glue_role_arn=<AWSGlueServiceRole>`.
4. [Optional]: Wenn für Ihr Profil keine Standardregion festgelegt ist, sollten Sie eine mit `region=us-east-1` hinzufügen und `us-east-1` durch die gewünschte Region ersetzen.

5. Speichern Sie die Konfiguration.

Weitere Informationen finden Sie unter Interaktive Sitzungen mit IAM.

Aktualisieren aus der Vorschau der interaktiven Sitzungen

Der Kernel wurde mit neuen Namen aktualisiert, als er mit Version 0.27 veröffentlicht wurde. Um die Vorschauversionen der Kernel zu bereinigen, führen Sie Folgendes von einem Terminal aus oder PowerShell aus.

> **Note**
>
> Wenn Sie Teil eines anderen sind AWS Glue Eine Vorschau, die ein benutzerdefiniertes Servicemodell erfordert, wird durch das Entfernen des Kernels das benutzerdefinierte Servicemodell entfernt.

```
# Remove Old Glue Kernels
jupyter kernelspec remove glue_python_kernel
jupyter kernelspec remove glue_scala_kernel

# Remove Custom Model
cd ~/.aws/models
rm -rf glue/
```

Interaktive Sitzungen mit SageMaker AI Studio verwenden

AWS Glue Interactive Sessions ist eine serverlose Apache Spark-Laufzeitumgebung auf Abruf, mit der Datenwissenschaftler und Ingenieure schnell Datenvorbereitungs- und Analyseanwendungen erstellen, testen und ausführen können. Sie können eine AWS Glue interaktive Sitzung starten, indem Sie ein Amazon SageMaker AI Studio Classic-Notizbuch starten.

Weitere Informationen finden Sie unter Vorbereiten von Daten mit AWS Glue interaktive Sitzungen.

Verwenden von interaktiven Sitzungen mit Microsoft Visual Studio Code

Voraussetzungen

- Installieren AWS Glue interaktive Sitzungen und überprüfen Sie, ob es mit Jupyter Notebook funktioniert.
- Downloaden und installieren Sie Visual Studio Code mit Jupyter. Details dazu finden Sie unter Jupyter Notebook in VS-Code.

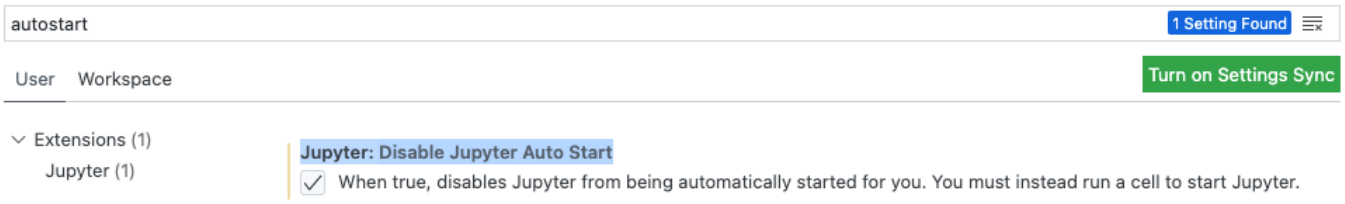
Um mit interaktiven Sitzungen zu beginnen mit VSCode

1. Deaktivieren Sie Jupyter AutoStart in VS Code.

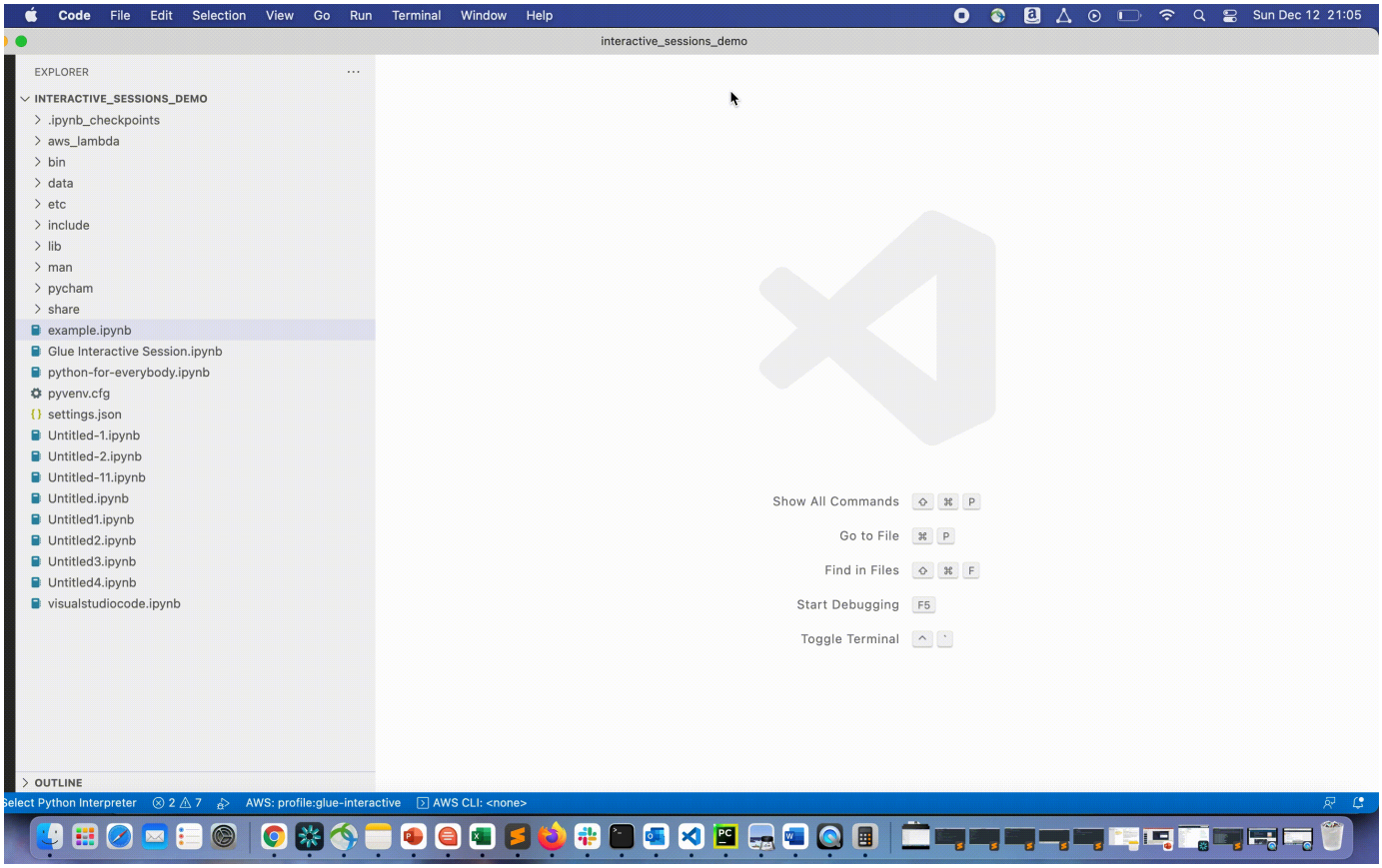
   In Visual Studio Code werden Jupyter-Kernel automatisch gestartet. Dies verhindert, dass Ihre Magics wirksam werden, da die Sitzung bereits gestartet wird. Um den automatischen Start unter Windows zu deaktivieren, gehen Sie zu Datei > Einstellungen > Erweiterungen > Jupyter > klicken Sie mit der rechten Maustaste auf Jupyter und wählen Sie dann Erweiterungseinstellungen.

   Gehen Sie unter MacOS zu Code > Einstellungen > Erweiterungen > Jupyter > klicken Sie mit der rechten Maustaste auf Jupyter und wählen Sie dann Erweiterungseinstellungen.

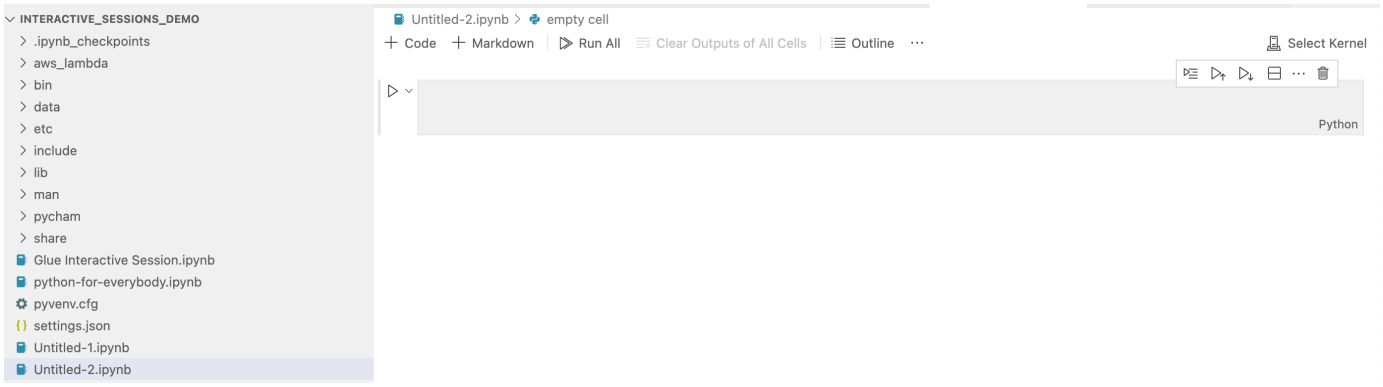
   Scrollen Sie nach unten, bis Sie Jupyter: Automatischer Start von Jupyter deaktivieren sehen. Aktivieren Sie das Kontrollkästchen „Falls wahr, wird Jupyter daran gehindert, automatisch für Sie gestartet zu werden. You must instead run a cell to start Jupyter.“ (Wenn „true“, wird Jupyter nicht automatisch gestartet. Sie müssen dann eine Zelle ausführen, um Jupyter zu starten.)



2. Gehen Sie auf File (Datei) > New File (Neue Datei) > Save (Speichern), um diese Datei mit dem Namen Ihrer Wahl als `.ipynb`-Erweiterung zu speichern, oder wählen Sie unter select a language (Sprache auswählen) jupyter aus und speichern Sie die Datei.



3. Doppelklicken Sie auf die Datei. Die Jupyter-Shell wird angezeigt und ein Notebook wird geöffnet.



4. Wenn Sie unter Windows zum ersten Mal eine Datei erstellen, ist standardmäßig kein Kernel ausgewählt. Klicken Sie auf Select Kernel (Kernel auswählen), um eine Liste der verfügbaren Kernel zu öffnen. Wählen Sie Glue PySpark.

   Wenn Sie unter macOS den PySparkGlue-Kernel nicht sehen, versuchen Sie die folgenden Schritte:

   1. Führen Sie eine lokale Jupyter-Sitzung aus, um die URL abzurufen.

Führen Sie beispielsweise den folgenden Befehl aus, um Jupyter Notebook zu starten.

```
jupyter notebook
```

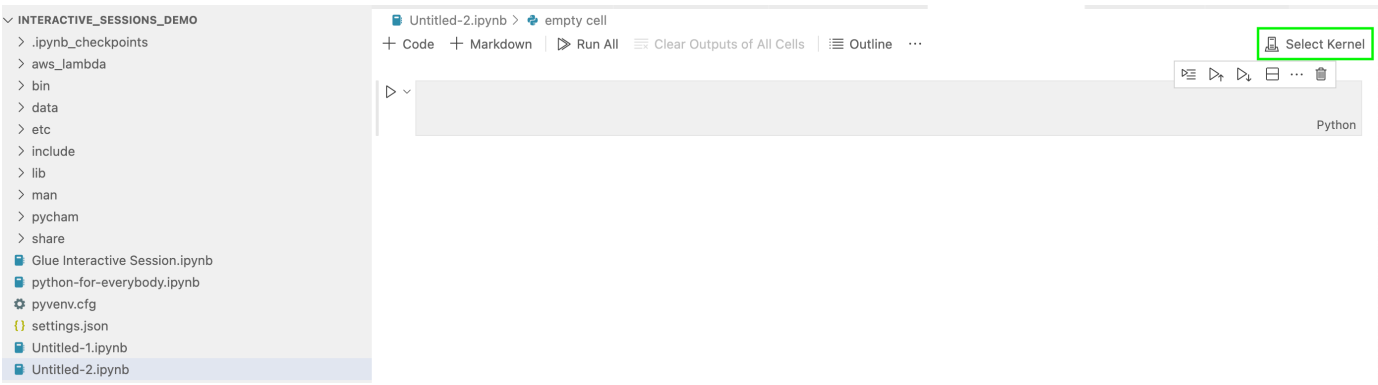
Wenn das Notebook zum ersten Mal ausgeführt wird, wird eine URL angezeigt, die wie folgt aussieht: `http://localhost:8888/?token=3398XXXXXXXXXXXXXXX`.

Kopieren Sie die URL.

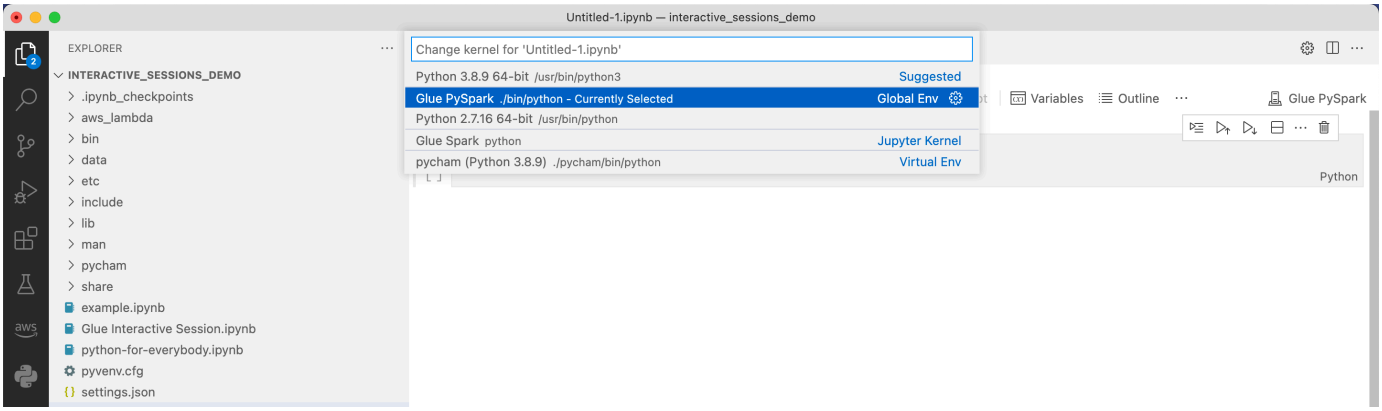
2. Klicken Sie in VS Code auf den aktuellen Kernel, dann auf Anderen Kernel auswählen... und dann auf Vorhandener Jupyter-Server…. Fügen Sie die URL ein, die Sie im obigen Schritt kopiert haben.

   Wenn Sie eine Fehlermeldung erhalten, lesen Sie das VS-Code-Jupyter-Wiki.

3. Wenn dies erfolgreich ist, wird der Kernel auf Glue gesetzt PySpark.



Wählen Sie den Glue PySpark - oder Glue Spark-Kernel (für Python bzw. Scala).



Wenn du nicht siehst AWS Glue PySpark und AWS Glue Bitte stellen Sie sicher, dass Sie die Spark-Kernel in der Drop-down-Liste installiert haben AWS Glue Kernel im obigen Schritt oder dass Ihre `python.defaultInterpreterPath` Einstellung in Visual Studio Code korrekt ist. Weitere Informationen finden Sie unter Python. defaultInterpreterPath Beschreibung der Einstellung.

5. Erstellen Sie eine AWS Glue Interaktive Sitzung. Fahren Sie mit dem Erstellen einer Sitzung auf die gleiche Weise wie in Jupyter Notebook fort. Geben Sie alle Magics oben in Ihrer ersten Zelle an und führen Sie eine Codeanweisung aus.

Interaktive Sitzungen mit IAM

In diesen Abschnitten werden Sicherheitsüberlegungen für interaktive AWS Glue-Sitzungen beschrieben.

Themen

- IAM-Prinzipale, die mit interaktiven Sitzungen verwendet werden
- Einrichten eines Client-Prinzipals
- Einrichten einer Laufzeitrolle
- Machen Sie Ihre Sitzung privat mit TagOnCreate
- Überlegungen zur IAM-Richtlinie

IAM-Prinzipale, die mit interaktiven Sitzungen verwendet werden

Sie verwenden zwei IAM-Prinzipale, die mit interaktiven AWS Glue-Sitzungen verwendet werden.

- Client-Prinzipal: Der Client-Prinzipal (entweder ein Benutzer oder eine Rolle) autorisiert API-Vorgänge für interaktive Sitzungen von einem AWS Glue-Client, der mit den identitätsbasierten Anmeldeinformationen des Prinzipals konfiguriert ist. Es könnte auch eine IAM-Rolle sein, mit der Sie normalerweise auf die AWS Glue-Konsole zugreifen. Dies könnte auch eine Rolle sein, die einem Benutzer in IAM zugewiesen wird, dessen Anmeldeinformationen für den Jupyter-Kernel verwendet werden AWS Command Line Interface, oder ein AWS Glue Client, der von den interaktiven Sitzungen verwendet wird.
- Laufzeitrolle: Die Laufzeitrolle ist eine IAM-Rolle, die der Client-Prinzipal an Operationen der Interactive Sessions API übergibt. AWS Glue verwendet diese Rolle, um Anweisungen in Ihrer

Sitzung auszuführen. Zum Beispiel könnte diese Rolle diejenige sein, die zum Ausführen von AWS Glue-ETL-Aufträgen verwendet wird.

Weitere Informationen finden Sie unter Einrichten einer Laufzeitrolle.

Einrichten eines Client-Prinzipals

Sie müssen eine Identitätsrichtlinie an den Client-Prinzipal anhängen, damit er die Interactive-Sessions-API aufrufen kann. Diese Rolle muss `iam:PassRole`-Zugriff auf die Ausführungsrolle haben, die Sie an die Interactive-Sessions-API übergeben würden, z. B. `CreateSession`. Sie können die AWSGlueConsoleFullAccessverwaltete Richtlinie beispielsweise einer IAM-Rolle zuordnen, sodass Benutzer in Ihrem Konto, an die die Richtlinie angehängt ist, auf alle in Ihrem Konto erstellten Sitzungen zugreifen können (z. B. Runtime Statement oder Cancel Statement).

Wenn Sie Ihre Sitzung schützen und sie nur für bestimmte IAM-Rollen privat machen möchten, z. B. für Rollen, die dem Benutzer zugeordnet sind, der die Sitzung erstellt hat, können Sie die tagbasierte Autorisierungssteuerung von AWS Glue Interactive Session verwenden. TagOnCreate Weitere Informationen finden Sie unter, wie Sie mit einer verwalteten Richtlinie, die Machen Sie Ihre Sitzung privat mit TagOnCreate auf einem Besitzertag basiert, Ihre Sitzung als privat kennzeichnen können. TagOnCreate Weitere Informationen zu identitätsbasierten Richtlinien finden Sie unter Identitätsbasierte Richtlinien für. AWS Glue

Einrichten einer Laufzeitrolle

Sie müssen eine IAM-Rolle an den CreateSession API-Vorgang übergeben, damit Anweisungen in interaktiven Sitzungen angenommen und ausgeführt werden könnenAWS Glue. Die Rolle sollte über dieselben IAM-Berechtigungen verfügen, die zum Ausführen eines gewöhnlichen AWS Glue-Auftrags erforderlich sind. Sie können beispielsweise mithilfe der AWSGlueServiceRoleRichtlinie, die es ermöglicht, Dienste in Ihrem Namen AWS Glue aufzurufen, eine AWS Servicerolle erstellen. Wenn Sie die AWS Glue-Konsole verwenden, erstellt sie automatisch eine Servicerolle in Ihrem Namen oder verwendet eine vorhandene Rolle. Sie können auch Ihre eigene IAM-Rolle erstellen und Ihre eigene IAM-Richtlinie anhängen, um ähnliche Berechtigungen zu gewähren.

Wenn Sie Ihre Sitzung schützen und sie nur für den Benutzer privat machen möchten, der die Sitzung erstellt hat, dann können Sie die Tag-Based Authorization Control von AWS Glue Interactive Session aufrufen TagOnCreate. Weitere Informationen finden Sie unter, wie Sie mit einer verwalteten Richtlinie, die Machen Sie Ihre Sitzung privat mit TagOnCreate auf einem Besitzertag basiert, Ihre Sitzung als privat kennzeichnen können. TagOnCreate Weitere Informationen zu

identitätsbasierten Richtlinien finden Sie unter Identitätsbasierte Richtlinien für Glue AWS. Wenn Sie die Ausführungsrolle selbst von der IAM-Konsole aus erstellen und Ihren Service mit dieser TagOnCreate Funktion als privat kennzeichnen möchten, gehen Sie wie folgt vor.

1. Erstellen Sie eine IAM-Rolle und legen Sie `Glue` als Rollentyp fest.
2. Hängen Sie diese AWS Glue verwaltete Richtlinie an: AwsGlueSessionUserRestrictedServiceRole
3. Stellen Sie dem Rollennamen den Richtliniennamen voran AwsGlueSessionUserRestrictedServiceRole. Sie können beispielsweise eine Rolle mit dem Namen AwsGlueSessionUserRestrictedServiceRole-myrole erstellen und eine AWS Glue verwaltete Richtlinie anhängen. AwsGlueSessionUserRestrictedServiceRole
4. Fügen Sie eine Vertrauensrichtlinie wie die folgende an, um AWS Glue zu ermöglichen, die Rolle zu übernehmen:

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
      ]
    }
  ]
}
```

Für einen Jupyter-Kernel für interaktive Sitzungen können Sie den `iam_role` Schlüssel in Ihrem Profil angeben. AWS Command Line Interface Weitere Informationen finden Sie unter Konfigurieren von Sitzungen mit ~/.aws/config . Wenn Sie über ein AWS Glue-Notebook mit interaktiven Sitzungen interagieren, können Sie die Ausführungsrolle im `%iam_role` Magic in der ersten Zelle, die Sie ausführen, übergeben.

Machen Sie Ihre Sitzung privat mit TagOnCreate

AWS Glue-interaktive Sitzungen unterstützen die Markierung und Tag-Based Authorization (TBAC) für interaktive Sitzungen als benannte Ressource. Zusätzlich zur Verwendung von TBAC TagResource und unterstützen AWS Glue interaktive Sitzungen die TagOnCreate Funktion UntagResource APIs, eine Sitzung nur während der Sitzungserstellung mit CreateSession einem bestimmten Tag zu „taggen“. Das bedeutet auch, dass diese Tags am entfernt werden DeleteSession, aka. UntagOnDelete

TagOnCreate bietet einen leistungsstarken Sicherheitsmechanismus, um Ihre Sitzung für den Ersteller der Sitzung privat zu machen. Sie können beispielsweise eine IAM-Richtlinie mit „owner“ RequestTag und dem Wert $ {aws:UserId} an einen Client-Principal (z. B. einen Benutzer) anhängen, um das Erstellen einer Sitzung nur zu ermöglichen, wenn auf Anfrage ein „owner“ -Tag mit dem entsprechenden Wert der userId des Anrufers als userID-Tag bereitgestellt wird. CreateSession Diese Richtlinie ermöglicht AWS Glue-interaktiven Sitzungen, eine Sitzungsressource zu erstellen und die Sitzung ausschließlich während der Erstellung mit dem userId-Tag zu markieren. Darüber hinaus können Sie den Zugriff (wie das Ausführen von Anweisungen) auf Ihre Sitzung nur auf den Ersteller (auch bekannt als Eigentümer-Tag mit dem Wert $ {aws:UserId}) beschränken, indem Sie der Ausführungsrolle, die Sie während der Sitzung übergeben haben, eine IAM-Richtlinie mit „owner“ ResourceTag anhängen CreateSession.

Um es Ihnen zu erleichtern, die TagOnCreate Funktion zu verwenden, mit der Sie eine Sitzung nur für den Sitzungsersteller sperren können, werden spezielle verwaltete Richtlinien und AWS Glue Servicerollen bereitgestellt.

Wenn Sie eine AWS Glue interaktive Sitzung mithilfe eines AssumeRole IAM-Prinzipals erstellen möchten (d. h. mithilfe von Anmeldeinformationen, die durch die Übernahme einer IAM-Rolle vergeben wurden) und Sie die Sitzung für den Ersteller privat machen möchten, verwenden Sie Richtlinien, die den bzw. entsprechen. AWSGlueSessionUserRestrictedNotebookPolicyAWSGlueSessionUserRestrictedNotebookServiceRole Diese Richtlinien ermöglichen esAWS Glue, $ {aws:PrincipalTag} zu verwenden, um den Besitzer-Tag-Wert zu extrahieren. Dazu müssen Sie ein UserID-Tag mit dem Wert $ {aws:UserId} übergeben, wie SessionTag in den Anmeldeinformationen für die Übernahme der Rolle. Siehe ID-Sitzungs-Tags. Wenn Sie eine EC2 Amazon-Instance mit einem Instance-Profil verwenden, das die Anmeldeinformationen verkauft, und Sie eine Sitzung erstellen oder mit der Sitzung von der EC2 Amazon-Instance aus interagieren möchten, müssen Sie ein UserID-Tag mit dem Wert $ {aws:UserId} übergeben, wie in den Anmeldeinformationen für die Rolle annehmen. SessionTag

Wenn Sie beispielsweise eine Sitzung mit einem AssumeRole IAM-Prinzipalnachweis erstellen und Ihren Service mit dieser Funktion als privat kennzeichnen möchten, gehen Sie wie folgt vor. TagOnCreate

1. Erstellen Sie eine Laufzeitrolle in der IAM-Konsole. Bitte hängen Sie diese AWS Glue verwaltete Richtlinie an AwsGlueSessionUserRestrictedNotebookServiceRoleund stellen Sie dem Rollennamen den Richtliniennamen voran. AwsGlueSessionUserRestrictedNotebookServiceRole Sie können beispielsweise eine Rolle mit dem Namen AwsGlueSessionUserRestrictedNotebookServiceRole-myrole erstellen und eine AWS Glue verwaltete Richtlinie anhängen. AwsGlueSessionUserRestrictedNotebookServiceRole
2. Fügen Sie eine Vertrauensrichtlinie wie folgt an, um AWS Glue zu ermöglichen, die obige Rolle zu übernehmen.

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": [
             "glue.amazonaws.com"
           ]
         },
         "Action": [
           "sts:AssumeRole"
          ]
       }
      ]
   }
   ```

3. Erstellen Sie eine weitere Rolle mit einem Präfix AwsGlueSessionUserRestrictedNotebookPolicyund fügen Sie die AWS Glue verwaltete Richtlinie an AwsGlueSessionUserRestrictedNotebookPolicy, um die Sitzung privat zu machen. Fügen Sie der Rolle, die Sie in Schritt 1 erstellt haben, zusätzlich zur verwalteten Richtlinie die folgende Inline-Richtlinie hinzuPassRole , um iam zuzulassen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::*:role/
AwsGlueSessionUserRestrictedNotebookServiceRole*"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

4. Fügen Sie eine Vertrauensrichtlinie wie folgt an, um dem obigen IAM-AWS Glue zu ermöglichen, die Rolle zu übernehmen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": [
                "glue.amazonaws.com"
            ]
        },
        "Action": [
```

```
            "sts:AssumeRole",
            "sts:TagSession"
        ]
    }]
}
```

> **Note**
>
> Optional können Sie eine einzelne Rolle (z. B. die Notebook-Rolle) verwenden und die beiden oben genannten verwalteten Richtlinien AwsGlueSessionUserRestrictedNotebookServiceRoleund AwsGlueSessionUserRestrictedNotebookPolicyanhängen. Fügen Sie außerdem die zusätzliche Inline-Richtlinie an, um die Übergabe Ihrer Rolle mit `iam:passrole` an AWS Glue zuzulassen. Abschließend fügen Sie die oben genannte Vertrauensrichtlinie an, um `sts:AssumeRole` und `sts:TagSession` zuzulassen.

AWSGlueSessionUserRestrictedNotebookPolicy

Die AWSGlue SessionUserRestrictedNotebookPolicy ermöglicht nur dann Zugriff auf die Erstellung einer AWS Glue interaktiven Sitzung von einem Notizbuch aus, wenn ein Tag-Schlüssel „Besitzer“ und ein Wert mit der AWS Benutzer-ID des Prinzipals (Benutzer oder Rolle) übereinstimmen. Weitere Informationen finden Sie unter Wo Richtlinienvariablen verwendet werden können. Diese Richtlinie ist dem Prinzipal (Benutzer oder Rolle) angefügt, der AWS Glue-Interactive-Sessions-Notebooks von AWS Glue Studio erstellt. Diese Richtlinie ermöglicht auch ausreichenden Zugriff auf das AWS Glue Studio Notizbuch, um mit den Ressourcen für AWS Glue Studio interaktive Sitzungen zu interagieren, die mit dem Tagwert „Besitzer“ erstellt wurden, der der AWS Benutzer-ID des Prinzipals entspricht. Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „owner“-Tags einer AWS Glue-Sitzungsressource, nachdem die Sitzung erstellt wurde.

AWSGlueSessionUserRestrictedNotebookServiceRole

Das AWSGlueSessionUserRestrictedNotebookServiceRolebietet ausreichend Zugriff auf das AWS Glue Studio Notizbuch, um mit den Ressourcen für AWS Glue interaktive Sitzungen zu interagieren, die mit dem Tagwert „owner“ erstellt wurden, der der AWS Benutzer-ID des Prinzipals (Benutzer oder Rolle) des Notizbuch-Erstellers entspricht. Weitere Informationen finden Sie unter Wo Richtlinienvariablen verwendet werden können. Diese Dienstrollenrichtlinie ist der Rolle zugeordnet,

die als magische Rolle an ein Notizbuch oder als Ausführungsrolle an die CreateSession API übergeben wird. Mit dieser Richtlinie kann auch nur dann eine AWS Glue interaktive Sitzung von einem Notizbuch aus erstellt werden, wenn der Tag-Schlüssel „Besitzer“ und der Wert mit der AWS Benutzer-ID des Prinzipals übereinstimmen. Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „owner“-Tags einer AWS Glue-Sitzungsressource, nachdem die Sitzung erstellt wurde. Diese Richtlinie umfasst auch Berechtigungen zum Schreiben und Lesen aus Amazon S3 S3-Buckets, zum Schreiben von CloudWatch Protokollen sowie zum Erstellen und Löschen von Tags für EC2 Amazon-Ressourcen, die von AWS Glue verwendet werden.

Ihre Sitzung mit Benutzer-Richtlinien privat stellen

Sie können die IAM-Rollen AWSGlueSessionUserRestrictedPolicy, die den einzelnen Benutzern in Ihrem Konto zugewiesen sind, zuordnen, um sie daran zu hindern, eine Sitzung nur mit einem Besitzer-Tag zu erstellen, dessen Wert ihrem eigenen $ {aws:UserId} entspricht. Anstatt das AWSGlueSessionUserRestrictedNotebookPolicyund zu verwenden, müssen AWSGlueSessionUserRestrictedNotebookServiceRoleSie ähnliche Richtlinien wie das und verwenden. AWSGlueSessionUserRestrictedPolicyAWSGlueSessionUserRestrictedServiceRole Weitere Informationen finden Sie unter Verwendung identitätsbasierter Richtlinien. Diese Richtlinie beschränkt den Zugriff auf eine Sitzung nur auf den Ersteller, die ${aws:userId} des Benutzers, der die Sitzung erstellt hat, mit einem Besitzer-Tag, der seine eigene ${aws:userId} trägt. Wenn Sie die Ausführungsrolle mithilfe der IAM-Konsole selbst erstellt haben, indem Sie die Schritte unter befolgenEinrichten einer Laufzeitrolle, dann fügen Sie nicht nur die AwsGlueSessionUserRestrictedPolicyverwaltete Richtlinie hinzu, sondern fügen Sie jedem Benutzer in Ihrem Konto auch die folgende Inline-Richtlinie hinzu, `iam:PassRole` um die zuvor erstellte Ausführungsrolle zu ermöglichen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:PassRole"
        ],
        "Resource": [
            "arn:aws:iam::*:role/AwsGlueSessionUserRestrictedServiceRole*"
        ],
        "Condition": {
```

```
            "StringLike": {
                "iam:PassedToService": [
                    "glue.amazonaws.com"
                ]
            }
        }
    }]
}
```

AWSGlueSessionUserRestrictedPolicy

Die AWSGlueSessionUserRestrictedPolicyermöglicht nur dann Zugriff auf die Erstellung einer AWS Glue interaktiven Sitzung mithilfe der CreateSession API, wenn ein Tag-Schlüssel „Eigentümer“ und ein Wert angegeben werden, der der AWS Benutzer-ID entspricht. Diese Identitätsrichtlinie ist an den Benutzer angehängt, der die CreateSession API aufruft. Diese Richtlinie ermöglicht auch die Interaktion mit den Ressourcen für AWS Glue interaktive Sitzungen, die mit einem „Eigentümer“ -Tag und einem Wert erstellt wurden, der ihrer AWS Benutzer-ID entspricht. Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „owner“-Tags einer AWS Glue-Sitzungsressource, nachdem die Sitzung erstellt wurde.

AWSGlueSessionUserRestrictedServiceRole

Die AWSGlueSessionUserRestrictedServiceRolebietet vollen Zugriff auf alle AWS Glue Ressourcen mit Ausnahme von Sitzungen und ermöglicht Benutzern, nur die interaktiven Sitzungen zu erstellen und zu verwenden, die dem Benutzer zugeordnet sind. Diese Richtlinie umfasst auch andere Berechtigungen, die für AWS Glue die Verwaltung von Glue-Ressourcen in anderen AWS Diensten erforderlich sind. Die Richtlinie ermöglicht auch das Hinzufügen von Tags zu AWS Glue Ressourcen in anderen AWS Diensten.

Überlegungen zur IAM-Richtlinie

Interaktive Sitzungen sind IAM-Ressourcen in AWS Glue. Da es sich um IAM-Ressourcen handelt, werden der Zugriff auf und die Interaktion mit einer Sitzung durch IAM-Richtlinien geregelt. Basierend auf den IAM-Richtlinien, die an einen Client-Prinzipal oder eine von einem Administrator konfigurierte Ausführungsrolle angehängt sind, kann ein Client-Prinzipal (Benutzer oder Rolle) neue Sitzungen erstellen und mit eigenen Sitzungen und anderen Sitzungen interagieren.

Wenn ein Administrator eine IAM-Richtlinie angehängt hat AWSGlueServiceRole , die beispielsweise AWSGlue ConsoleFullAccess den Zugriff auf alle AWS Glue Ressourcen in diesem Konto ermöglicht,

kann ein Client-Principal zusammenarbeiten. Beispielsweise kann ein Benutzer mit Sitzungen interagieren, die von anderen Benutzern erstellt wurden, wenn Richtlinien dies zulassen.

Wenn Sie eine Richtlinie konfigurieren möchten, die auf Ihre spezifischen Bedürfnisse zugeschnitten ist, finden Sie in der IAM-Dokumentation zur Konfiguration von Ressourcen für eine Richtlinie. Um beispielsweise Sitzungen zu isolieren, die einem Benutzer gehören, können Sie die von AWS Glue interaktiven Sitzungen unterstützte TagOnCreate Funktion verwenden. Siehe Machen Sie Ihre Sitzung privat mit TagOnCreate .

Interaktive Sitzungen unterstützen die Einschränkung der Sitzungserstellung basierend auf bestimmten VPC-Bedingungen. Siehe Steuern von Richtlinien, die Einstellungen über Bedingungsschlüssel steuern.

Konfigurieren von AWS Glue-interaktiven Sitzungen für Jupyter und AWS Glue Studio-Notebooks

Einführung in Jupyter Magics

Jupyter Magics sind Befehle, die am Anfang einer Zelle oder als ganzer Zellinhalt ausgeführt werden können. Zeilen-Magics beginnen mit `%` und Zellen-Magics mit `%%`. Zeilen-Magics wie `%region` und `%connections` können wie im folgenden Beispiel mit mehreren Magics in einer Zelle oder mit Code im Zellinhalt ausgeführt werden.

```
%region us-east-2
%connections my_rds_connection
dy_f = glue_context.create_dynamic_frame.from_catalog(database='rds_tables',
 table_name='sales_table')
```

Zell-Magics müssen die gesamte Zelle verwenden und der Befehl kann sich über mehrere Zeilen erstrecken. Ein Beispiel für `%%sql` sehen Sie unten.

```
%%sql
select * from rds_tables.sales_table
```

Unterstützte Magics in AWS Glue-interaktiven Sitzungen für Jupyter

Im Folgenden finden Sie Magics, die Sie in AWS Glue Interactive Sessions für Jupyter Notebooks verwenden können.

Sessions Magics

Name	Typ	Beschreibung
`%help`	–	Gibt eine Liste von Beschreibungen und Eingabetypen für alle magischen Befehle zurück.
`%profile`	String	Geben Sie in Ihrer AWS Konfiguration ein Profil an, das als Anbieter für Anmeldein formationen verwendet werden soll.
`%region`	String	Geben Sie den AWS-Region; an, in dem eine Sitzung initialisiert werden soll. Standardwert aus `~/.aws/configure.` Beispiel: `%region us-west-1`
`%idle_timeout`	Int	Die Anzahl der Minuten von Inaktivit ät, nach denen eine Zeitüberschreitung für eine Sitzung auftritt, nachdem eine Zelle ausgeführt wurde. Der Standardw ert für die Zeitüberschreitung für Spark-ETL-Sitzungen ist der Standardw ert für die Zeitüberschreitung, 2 880 Minuten (48 Stunden). Informationen zu anderen Sitzungstypen finden Sie in der Dokumentation für diesen Sitzungstyp. Beispiel: `%idle_timeout 3000`
`%session_id`	–	Gibt die Sitzungs-ID für die laufende Sitzung zurück.
`%session_id_prefix`	String	Definieren Sie eine Zeichenfolge, die allen Sitzungen vorangeht, IDs im Format

Name	Typ	Beschreibung
		[session_id_prefix] - [session_id]. Wenn keine Sitzungs-ID angegeben wird, wird eine zufällige UUID generiert. Dieses Magic wird nicht unterstützt, wenn Sie ein Jupyter Notebook in AWS Glue Studio ausführen. Beispiel: `%session_id_prefix 001`
`%status`		Gibt den Status der aktuellen AWS Glue-Sitzung zurück, einschließlich Dauer, Konfiguration und ausführende(r) Benutzer/Rolle.
`%stop_session`		Beenden Sie die aktuelle Sitzung.
`%list_sessions`		Listet alle derzeit ausgeführten Sitzungen nach Name und ID auf.
`%session_type`	String	Legt den Sitzungstyp auf Streaming, ETL oder Ray fest. Beispiel: `%session_type Streaming`
`%glue_version`	String	Die Version von AWS Glue, die von dieser Sitzung verwendet werden soll. Beispiel: `%glue_version 3.0`

Magics für die Auswahl von Auftragstypen

Name	Typ	Beschreibung
`%streaming`	String	Ändert den Sitzungstyp in AWS Glue Streaming.
`%etl`	String	Ändert den Sitzungstyp in AWS Glue ETL.

Name	Typ	Beschreibung
%glue_ray	String	AWS Glue Ändert den Sitzungstyp auf für Ray. Weitere Informationen finden Sie unter Magics, die von interaktiven AWS Glue Ray-Sitzungen unterstützt werden.

AWS Glue for Spark config magics

Das `%%configure`-Magic ist ein JSON-formatiertes Wörterbuch, das alle Konfigurationsparameter für eine Sitzung enthält. Jeder Parameter kann hier oder durch einzelne Magics angegeben werden.

Name	Typ	Beschreibung
`%%configure`	Dictionary	Geben Sie ein JSON-formatiertes Wörterbuch an, das aus allen Konfigura tionsparametern für eine Sitzung besteht. Jeder Parameter kann hier oder durch einzelne Magics angegeben werden. Eine Liste mit Parametern und Anwendung sbeispielen finden Sie `%%configure` unter%%configure cell magic arguments.
`%iam_role`	String	Gibt den ARN einer IAM-Rolle an, mit der Ihre Sitzung ausgeführt wird. Standardwert aus ~/.aws/configure. Beispiel: `%iam_role AWSGlueSe rviceRole`
`%number_of_workers`	Int	Die Anzahl der Worker eines definierten worker_type, die zugewiesen werden, wenn ein Auftrag ausgeführt wird. `worker_type` muss ebenfalls festgelegt werden. Der Standardwert für `number_of _workers` ist 5.

Name	Typ	Beschreibung
		Beispiel: `%number_of_workers 2`
`%additional_python_modules`	Auflisten	Durch Kommas getrennte Liste zusätzlicher Python-Module, die in Ihren Cluster aufgenommen werden sollen (kann von PyPI oder S3 stammen). Beispiel: `%additional_python_modules pandas, numpy .`

Name	Typ	Beschreibung

`%%tags`	String	Fügt einer Sitzung Tags hinzu. Geben Sie die Tags in geschweiften Klammern { } an. Jedes Tag-Namenspaar wird in Klammern („“) eingeschlossen und durch ein Komma (,) getrennt.
		
		`%%tags`
		`{"billing":"Data-Platform",`
		`"team":"analytics"}`
		
		Nutzen Sie dieses `%status` Magic, um mit der Sitzung verknüpfte Tags anzuzeigen.
		
		`%status`
		
		`Session ID: <sessionId>`
		`Status: READY`
		`Role: <example-role>`
		`CreatedOn: 2023-05-26 11:12:17.`
		`056000-07:00`
		`GlueVersion: 3.0`
		`Job Type: glueetl`
		`Tags: {'owner':'example-owner',`
		`'team':'analytics', 'billing'`
		`:'Data-Platform'}`
		`Worker Type: G.4X`
		`Number of Workers: 5`
		`Region: us-west-2`
		`Applying the following default`
		`arguments:`
		`--glue_kernel_version 0.38.0`
		`--enable-glue-datacatalog true`
		`Arguments Passed: ['--glue_`
		`kernel_version: 0.38.0', '--`
		`enable-glue-datacatalog: true']`

Name	Typ	Beschreibung

`%%assume_role`	Dictionary	Geben Sie ein JSON-formatiertes Wörterbuch oder eine IAM-Rollen-ARN-Zeichenfolge an, um eine Sitzung für den kontoübergreifenden Zugriff zu erstellen.
		
		Beispiel mit ARN:
		
		`%%assume_role`
		`{`
		`  'arn:aws:iam::XXXXXXXXXXXX:role/AWSGlueServiceRole'`
		`}`
		
		Beispiel mit Anmeldeinformationen:
		
		`%%assume_role`
		`{{`
		`  "aws_access_key_id" = "XXXXXXXXXXXX",`
		`  "aws_secret_access_key" = "XXXXXXXXXXXX",`
		`  "aws_session_token" = "XXXXXXXXXXXX"`
		`}}`

%%configure cell magic arguments

Das `%%configure`-Magic ist ein JSON-formatiertes Wörterbuch, das alle Konfigurationsparameter für eine Sitzung enthält. Jeder Parameter kann hier oder durch einzelne Magics angegeben werden. Nachfolgend finden Sie Beispiele für Argumente, die von dem `%%configure`-Zellen-Magic unterstützt werden. Verwenden Sie das `--` Präfix für Ausführungsargumente, die für den Job angegeben wurden. Beispiel:

```
%%configure
```

```
{
   "--user-jars-first": "true",
   "--enable-glue-datacatalog": "false"
}
```

Weitere Informationen zu Auftragsparametern finden Sie unterAuftragsparameter.

Konfiguration der Sitzung

Parameter	Typ	Beschreibung

`max_retries`	Int	Die maximale Anzahl der Wiederhol ungsversuche für diesen Auftrag, wenn er fehlschlägt.
		`%%configure`
		`{`
		`  "max_retries": "0"`
		`}`

`max_concurrent_runs`	Int	Die maximale Anzahl der gleichzeitigen Ausführungen, die für einen Auftrag zulässig sind.
		Beispiel:
		`%%configure`
		`{`
		`  "max_concurrent_runs": "3"`
		`}`

Sitzungsparameter

Parameter	Typ	Beschreibung

`--enable-spark-ui`	Boolesch	Aktivieren Sie die Spark-Benutzerober fläche zum Überwachen und Debuggen von AWS Glue-ETL-Aufträgen.
		`%%configure`
		`{`
		`  "--enable-spark-ui": "true"`
		`}`

`--spark-event-logs-path`	String	Gibt einen Amazon-S3-Pfad an. Bei Verwendung des Spark UI-Überwachungs-Features.
		Beispiel:
		`%%configure`
		`{`
		`  "--spark-event-logs-path":`
		`"s3://path/to/event/logs/"`
		`}`

`--script_location`	String	Gibt den S3-Pfad zu einem Skript an, das einen Auftrag ausführt.
		Beispiel:
		`%%configure`
		`{`
		`  "script_location": "s3://new-folder-here"`
		`}`

`--SECURITY_CONFIGURATION`	String	Der Name einer AWS Glue Sicherheitskonfiguration

Parameter	Typ	Beschreibung

		Beispiel:
		
		`%%configure`
		`{`
		`"--security_configuration": { "encryption_type": "kms", "kms_key_id": "YOUR_KMS _KEY_ARN" }`
		`}`

`--job-language`	String	Die Skript-Programmiersprache. Akzeptiert den Wert „scala“ oder „python“. Die Standardeinstellung ist „python“.
		
		Beispiel:
		
		`%%configure`
		`{`
		`"--job-language": "scala"`
		`}`

`--class`	String	Die Scala-Klasse, die als Einstiegspunkt für Ihr Scala-Skript dient. Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`"--class": "className"`
		`}`

Parameter	Typ	Beschreibung

`--user-jars-first`	Boolesch	Priorisiert die zusätzlichen JAR-Datei en des Kunden im Klassenpfad. Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--user-jars-first": "true"`
		`}`

`--use-postgres-driver`	Boolesch	Priorisiert den Postgres-JDBC-Treiber im Klassenpfad, um einen Konflikt mit dem JDBC-Treiber zu vermeiden. Amazon Redshift Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--use-postgres-driver": "true"`
		`}`

Parameter	Typ	Beschreibung

`--extra-files`	List(string)	Die Amazon-S3-Pfade zu zusätzlichen Dateien, z. B. Konfigurationsdateien, die AWS Glue vor der Ausführung in das Arbeitsverzeichnis Ihres Skripts kopiert.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--extra-files": "s3://path/to/additional/files/"`
		`}`

`--job-bookmark-option`	String	Steuert die Darstellung eines Auftrags-Lesezeichens. Akzeptiert den Wert '', '' oder ''. job-bookmark-enable job-bookmark-disable job-bookmark-pause Die Standardeinstellung ist 'job-bookmark-disable'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--job-bookmark-option": "job-bookmark-enable"`
		`}`

Parameter	Typ	Beschreibung

`--TempDir`	String	Gibt einen Amazon-S3-Pfad zu einem Bucket an, der als temporäres Verzeichn is für den Auftrag verwendet werden kann. Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--TempDir": "s3://path/to/temp`
		`/dir"`
		`}`

`--enable-s3-parquet-optimized-committer`	Boolesch	Aktiviert den für EMRFS Amazon S3 optimierten Committer zum Schreiben von Parquet-Daten in Amazon S3. Die Standardeinstellung ist 'true'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-s3-parquet-optimi`
		`zed-committer": "false"`
		`}`

Parameter	Typ	Beschreibung

`--enable-rename-algorithm-v2`	Boolesch	Setzt die Version des EMRFS-Umbenennungsalgorithmus auf Version 2. Die Standardeinstellung ist 'true'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-rename-algorithm-v2": "true"`
		`}`

`--enable-glue-datacatalog`	Boolesch	Ermöglicht Ihnen die Verwendung des AWS Glue Data Catalog als Apache-Spark-Hive-Metaspeicher.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-glue-datacatalog": "true"`
		`}`

`--enable-metrics`	Boolesch	Ermöglicht die Erfassung von Metriken zur Auftragsprofilerstellung für die Auftragsausführung. Standard ist 'false'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-metrics": "true"`
		`}`

Parameter	Typ	Beschreibung

`--enable-continuous-cloudwatch-log`	Boolesch	Aktiviert die kontinuierliche Echtzeitp rotokollierung für AWS Glue-Aufträge. Standard ist 'false'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-continuous-cloudw atch-log": "true"`
		`}`

`--enable-continuous-log-filter`	Boolesch	Gibt einen Standardfilter oder keinen Filter an, wenn Sie einen Auftrag erstellen oder bearbeiten, der für die kontinuierliche Protokollierung aktiviert ist. Die Standarde instellung ist 'true'.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--enable-continuous-log-fi lter": "true"`
		`}`

Parameter	Typ	Beschreibung

`--continuous-log-stream-prefix`	String	Gibt ein benutzerdefiniertes Amazon CloudWatch Protokollstream-Präfix für einen Job an, der für die kontinuierliche Protokollierung aktiviert ist. Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--continuous-log-stream-prefix": "prefix"`
		`}`

`--continuous-log-conversionPattern`	String	Gibt ein benutzerdefiniertes Konvertierungsprotokollmuster für einen Auftrag an, der für die kontinuierliche Protokollierung aktiviert ist. Die Standardeinstellung ist null.
		
		Beispiel:
		
		`%%configure`
		`{`
		`  "--continuous-log-conversionPattern": "pattern"`
		`}`

Parameter	Typ	Beschreibung

`--conf`	String	Sie steuert die Spark-Konfiguratio nsparameter. Sie ist für fortschrittliche Anwendungsfälle. Wird `--conf` vor jedem Parameter verwendet. Beispiel:
		`%%configure`
		`{`
		`"--conf": "spark.hadoop.hive`
		`.metastore.glue.catalogid=1`
		`23456789012 --conf hive.meta`
		`store.client.factory.class=`
		`com.amazonaws.glue.catalog.`
		`metastore.AWSGlueDataCatalo`
		`gHiveClientFactory --conf`
		`hive.metastore.schema.verif`
		`ication=false"`
		`}`

timeout	Int	Legt fest, wie lange die Spark-Sitzung maximal warten soll, bis eine Anweisung abgeschlossen ist, bevor sie beendet wird.
		`%%configure`
		`{`
		`"timeout": "30"`
		`}`

auto-scaling	Boolesch	Legt fest, ob die auto-scaling verwendet werden soll oder nicht.
		`%%configure`
		`{`
		`"--enable-auto-scaling": "true"`
		`}`

Magic von Spark-Aufträgen (ETL und Streaming)

Name	Typ	Beschreibung
`%worker_type`	String	Standard, G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X oder R.8X. `number_of_workers` muss auch eingestellt sein. Der Standardwert für „worker_type“ ist G.1X.
`%connections`	Auflisten	Geben Sie eine kommagetrennte Liste der Verbindungen an, die in der Sitzung verwendet werden sollen. Beispiel: `%connections my_rds_connection dy_f = glue_context.create_dynamic _frame.from_catalog(databas e='rds_tables', table_nam e='sales_table')`
`%extra_py_files`	Auflisten	Durch Kommas getrennte Liste mit zusätzlichen Python-Dateien von Amazon S3.
`%extra_jars`	Auflisten	Durch Kommas getrennte Liste mit zusätzlichen Jars, die in den Cluster aufgenommen werden sollen.
`%spark_conf`	String	Geben Sie benutzerdefinierte Spark-Konfigurationen für Ihre Sitzung an. Beispiel, `%spark_conf spark.ser ializer=org.apache.spark.se rializer.KryoSerializer` .

Magics für Ray-Aufträgen

Name	Typ	Beschreibung
`%min_workers`	Int	Die Mindestanzahl von Workern, die einem Ray-Auftrag zugewiesen werden. Standard: 1 Beispiel: `%min_workers 2`
`%object_memory_head`	Int	Der Prozentsatz des freien Speichers auf dem Hauptknoten der Instance nach einem Warmstart. Minimum: 0. Maximum: 100. Beispiel: `%object_memory_head 100`
%object_memory_worker	Int	Der Prozentsatz des freien Arbeitssp eichers auf den Instance-Worker-Knoten nach einem Warmstart. Minimum: 0. Maximum: 100. Beispiel: `%object_memory_worker 100`

Action Magics

Name	Typ	Beschreibung
`%%sql`	String	Führt SQL-Code aus. Alle Zeilen nach dem ersten `%%sql`-Magic werden als Teil des SQL-Codes übergeben. Beispiel: `%%sql select * from rds_tables.sales_table`
`%matplot`	Matplotlib-Abbildung	Visualisieren Sie Ihre Daten mit der Matplotlib-Bibliothek.

Name	Typ	Beschreibung

		Beispiel:
		`import matplotlib.pyplot as plt`
		
		`# Set X-axis and Y-axis values`
		`x = [5, 2, 8, 4, 9]`
		`y = [10, 4, 8, 5, 2]`
		
		`# Create a bar chart`
		`plt.bar(x, y)`
		
		`# Show the plot`
		`%matplot plt`

`%plotly`	Plotly-Abbildung	Visualisieren Sie Ihre Daten mit der Plotly-Bibliothek.
		
		Beispiel:
		`import plotly.express as px`
		
		`#Create a graphical figure`
		`fig = px.line(x=["a","b","c"],`
		` y=[1,3,2], title="sample`
		` figure")`
		
		`#Show the figure`
		`%plotly fig`

Benennen von Sitzungen

AWS Glueinteraktive Sitzungen sind AWS Ressourcen und benötigen einen Namen. Namen sollten für jede Sitzung eindeutig sein und können von Ihren IAM-Administratoren eingeschränkt werden. Weitere Informationen finden Sie unter Interaktive Sitzungen mit IAM. Der Jupyter-Kernel generiert automatisch eindeutige Sitzungsnamen für Sie. Es gibt jedoch zwei Möglichkeiten, um Sitzungen manuell zu benennen:

1. Verwenden Sie die AWS Command Line Interface Konfigurationsdatei unter`~.aws/config`. Siehe AWS Config einrichten mit dem AWS Command Line Interface.
2. Verwendung der `%session_id_prefix`-Magics. Siehe Unterstützte Magics in AWS Glue-interaktiven Sitzungen für Jupyter .

Ein Sitzungsname wird wie folgt generiert:

- Wenn das Präfix und session_id angegeben werden: Der Sitzungsname lautet {prefix}-{UUID}.
- Wenn nichts angegeben wird: Der Sitzungsname lautet {UUID}.

Wenn Sie Sitzungsnamen als Präfix angeben, können Sie Ihre Sitzung erkennen, wenn Sie sie in der AWS CLI OR-Konsole auflisten.

Angeben einer IAM-Rolle für interaktive Sitzungen

Sie müssen eine AWS Identity and Access Management (IAM) -Rolle angeben, die mit AWS Glue ETL-Code verwendet werden soll, den Sie mit interaktiven Sitzungen ausführen.

Die Rolle muss über dieselben IAM-Berechtigungen verfügen, die zum Ausführen von AWS Glue-Aufträgen erforderlich sind. Weitere Informationen zum Erstellen einer Rolle für AWS Glue-Aufträge und interaktive Sitzungen finden Sie unter Erstellen einer IAM-Rolle für AWS Glue.

IAM-Rollen können auf zwei Arten angegeben werden:

- Verwenden Sie die AWS Command Line Interface Konfigurationsdatei unter `~.aws/config` (Empfohlen). Weitere Informationen finden Sie unter Konfigurieren von Sitzungen mit ~/.aws/config .

  > **Note**
  >
  > Wenn die Magic-Anweisung `%profile` verwendet wird, wird die Konfiguration für `glue_iam_role` dieses Profils berücksichtigt.

- Verwendung der Magic-Anweisung „%iam_role“. Weitere Informationen finden Sie unter Unterstützte Magics in AWS Glue-interaktiven Sitzungen für Jupyter .

Konfigurieren von Sitzungen mit benannten Profilen

AWS Glueinteraktive Sitzungen verwenden dieselben Anmeldeinformationen wie AWS Command Line Interface oder boto3, und interaktive Sitzungen berücksichtigen und arbeiten mit benannten Profilen, wie sie in `~/.aws/config` (Linux und macOS) oder `%USERPROFILE%\.aws\config` (Windows) zu AWS CLI finden sind. Weitere Informationen finden Sie unter Verwendung benannter Profile.

Interaktive Sitzungen nutzen benannte Profile, indem sie die Angabe der AWS Glue-Servicerolle und des Sitzungs-ID-Präfix in einem Profil zulassen. Um eine Profilrolle zu konfigurieren, fügen Sie eine Zeile für den `iam_role` Schlüssel and/or `session_id_prefix` zu Ihrem benannten Profil hinzu, wie unten gezeigt. Für sind `session_id_prefix` keine Anführungszeichen erforderlich. Wenn Sie beispielsweise einen `session_id_prefix` hinzufügen möchten, geben Sie den Wert von `session_id_prefix=myprefix` ein.

```
[default]
region=us-east-1
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRole>
session_id_prefix=<prefix_for_session_names>

[user1]
region=eu-west-1
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRoleUser1>
session_id_prefix=<prefix_for_session_names_for_user1>
```

Wenn Sie eine benutzerdefinierte Methode zum Generieren von Anmeldeinformationen nutzen, können Sie Ihr Profil auch dafür konfigurieren, `credential_process`-Parameter in Ihrer `~/.aws/config`-Datei zu verwenden. Zum Beispiel:

```
[profile developer]
region=us-east-1
credential_process = "/Users/Dave/generate_my_credentials.sh" --username helen
```

Weitere Informationen zur Beschaffung von Anmeldeinformationen mithilfe der `credential_process`-Parameter finden Sie hier: Beschaffung von Anmeldeinformationen mit einem externen Prozess.

Wenn eine Region oder `iam_role` in dem von Ihnen verwendeten Profil nicht festgelegt sind, müssen Sie sie mit den Magic-Anweisungen `%region` und `%iam_role` in der ersten Zelle angeben, die Sie ausführen.

Ein Skript oder ein Notizbuch in einen AWS Glue Job umwandeln

Es gibt zwei Möglichkeiten, ein Skript oder ein Notizbuch in ein umzuwandeln AWS Glue Beruf:

- Sie können Ihre Jupyter-Notebook-Dokumentdatei (`.ipynb`) mit nbconvert in eine `.py`-Datei umwandeln. Weitere Informationen finden Sie unter nbconvert: Konvertieren von Notebooks in andere Formate.
- Laden Sie die Datei hoch auf AWS Glue Studio Notizbücher.
  - Im AWS Glue Studio Konsole, wählen Sie Jobs aus dem Navigationsmenü.
  - Wählen Sie im Bereich Create job (Auftrag erstellen) die Option Jupyter Notebook aus.
  - Gehen Sie im Bereich Options (Optionen) auf Upload and edit an existing notebook (Bestehendes Notebook hochladen und bearbeiten).
  - Wählen Sie Choose file (Datei auswählen) aus, um eine `.ipynb`-Datei hochzuladen.

Arbeiten mit Streaming-Vorgängen in AWS Glue interaktive Sitzungen

Wechseln des Streaming-Sitzungstyps

Benutze die AWS Glue Magic zur Konfiguration interaktiver Sitzungen,`%streaming`, um den Job zu definieren, den Sie gerade ausführen, und um eine interaktive Streaming-Sitzung zu initialisieren.

Sampling des Eingabestreams für die interaktive Entwicklung

Ein Tool, das wir entwickelt haben, um das interaktive Erlebnis zu verbessern AWS Glue Bei interaktiven Sitzungen handelt es sich `GlueContext` um eine neue Methode, mit der Sie einen Snapshot eines Streams in einer statischen Datei abrufen können DynamicFrame.

`GlueContext`ermöglicht es Ihnen, Ihren Arbeitsablauf zu überprüfen, zu interagieren und zu implementieren.

Anhand der `GlueContext`-Klassen-Instance können Sie die Methode `getSampleStreamingDynamicFrame` finden. Erforderliche Argumente für diese Methode sind:

- `dataFrame`: Das Spark-Streaming DataFrame
- `options`: Verfügbare Optionen siehe unten

Verfügbare Optionen:

- windowSize: Dies wird auch als „Microbatch Duration“ bezeichnet. Dieser Parameter bestimmt, wie lange eine Streaming-Abfrage wartet, nachdem der vorherige Batch ausgelöst wurde. Der Parameterwert muss kleiner sein als `pollingTimeInMs`.
- pollingTimeInMs: Die Gesamtdauer, über die die Methode ausgeführt wird. Sie löst mindestens einen Microbatch aus, um Beispieldatensätze aus dem Eingabestream abzurufen.
- recordPollingLimit: Dieser Parameter hilft Ihnen, die Gesamtzahl der Datensätze zu begrenzen, die Sie aus dem Stream abfragen werden.
- (Optional) Sie können auch `writeStreamFunction` verwenden, um diese benutzerdefinierte Funktion auf jede Datensatz-Sampling-Funktion anzuwenden. Beispiele in Scala und Python finden Sie unten.

Scala

```
val sampleBatchFunction = (batchDF: DataFrame, batchId: Long) => {//Optional but
 you can replace your own forEachBatch function here}
val jsonString: String = s"""{"pollingTimeInMs": "10000", "windowSize": "5
 seconds"}"""
val dynFrame = glueContext.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF,
 JsonOptions(jsonString), sampleBatchFunction)
dynFrame.show()
```

Python

```
def sample_batch_function(batch_df, batch_id):
       //Optional but you can replace your own forEachBatch function here
```

```
options = {
            "pollingTimeInMs": "10000",
            "windowSize": "5 seconds",
        }
glue_context.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF, options,
 sample_batch_function)
```

> **Note**
>
> Wenn der `DynFrame` aus dem Sampling leer ist, kann das mehrere Gründe haben:
>
> - Die Streaming-Quelle ist auf „Neueste“ eingestellt und während des Sampling-Zeitraums wurden keine neuen Daten aufgenommen.
> - Die Abfragezeit reicht nicht aus, um die aufgenommenen Datensätze zu verarbeiten. Daten werden erst angezeigt, wenn der gesamte Batch verarbeitet wurde.

Ausführen von Streaming-Anwendungen in interaktive Sitzungen

In AWS Glue interaktive Sitzungen, die Sie ausführen können unter AWS Glue Streaming-Anwendung, wie Sie eine Streaming-Anwendung erstellen würden in AWS Glue Konsole. Da interaktive Sitzungen sitzungsbasiert sind, führen Ausnahmen in der Laufzeit nicht dazu, dass die Sitzung beendet wird. Wir haben jetzt den zusätzlichen Vorteil, Ihre Batch-Funktion iterativ entwickeln zu können. Zum Beispiel:

```
def batch_function(data_frame, batch_id):
    log.info(data_frame.count())
    invalid_method_call()
glueContext.forEachBatch(frame=streaming_df, batch_function = batch_function, options =
 {**})
```

Im obigen Beispiel haben wir eine ungültige Verwendung einer Methode aufgenommen, und zwar im Gegensatz zu regulären AWS Glue Jobs, die die gesamte Anwendung beenden, der Codierungskontext und die Definitionen des Benutzers bleiben vollständig erhalten und die Sitzung ist weiterhin betriebsbereit. Sie müssen keinen Bootstrap für einen neuen Cluster durchführen und die gesamte vorangegangene Transformation erneut ausführen. So können Sie sich auf die schnelle

Iteration Ihrer Batch-Funktionsimplementierungen konzentrieren, um die gewünschten Ergebnisse zu erzielen.

Beachten Sie, dass Interactive Sessions jede Anweisung blockierend auswertet, sodass die Sitzung immer nur eine Anweisung nach der anderen ausführt. Da Streaming-Abfragen kontinuierlich ausgeführt werden und niemals enden, können Sitzungen mit aktiven Streaming-Abfragen keine Follow-up-Anweisungen verarbeiten, ohne unterbrochen zu werden. Sie können den Unterbrechungsbefehl direkt in Jupyter Notebook ausgeben und unser Kernel wird den Abbruch für Sie vornehmen.

Das folgende Beispiel enthält eine Abfolge von Anweisungen, die auf die Ausführung warten:

```
Statement 1:
      val number = df.count()
      #Spark Action with deterministic result
      Result: 5

Statement 2:
      streamingQuery.start().awaitTermination()
      #Spark Streaming Query that will be executing continously
      Result: Constantly updated with each microbatch

Statement 3:
      val number2 = df.count()
      #This will not be executed as previous statement will be running indefinitely
```

Lokales Entwickeln und Testen von AWS Glue-Jobskripten

Wenn Sie Ihre AWS Glue for Spark-Jobskripte entwickeln und testen, stehen Ihnen mehrere Optionen zur Verfügung:

- AWS Glue Studio-Konsole
  - Visual editor (Visueller Editor)
  - Skript-Editor
  - AWS Glue Studio Notizbuch
- Interaktive Sitzungen
  - Jupyter Notebook

- Docker-Image
  - Lokale Entwicklung
  - Remote-Entwicklung

Sie können eine der oben genannten Optionen entsprechend Ihren Anforderungen auswählen.

Wenn Sie keinen Code oder weniger Code-Erfahrung bevorzugen, ist der Visual Editor von AWS Glue Studio eine gute Wahl.

Wenn Sie ein interaktives Notizbucherlebnis bevorzugen, ist das AWS Glue Studio-Notizbuch eine gute Wahl. Weitere Informationen finden Sie unter Notizbücher mit AWS Glue Studio und AWS Glue verwenden. Wenn Sie Ihre eigene lokale Umgebung nutzen möchten, sind interaktive Sitzungen eine gute Wahl. Weitere Informationen finden Sie unter Interaktive Sitzungen mit AWS Glue verwenden.

Wenn Sie local/remote Entwicklungserfahrung bevorzugen, ist das Docker-Image eine gute Wahl. Auf diese Weise können Sie AWS Glue for Spark-Jobskripte entwickeln und testen, wo immer Sie möchten, ohne dass AWS Glue-Kosten anfallen.

Wenn Sie lokale Entwicklung ohne Docker bevorzugen, ist die lokale Installation des AWS Glue ETL-Bibliotheksverzeichnisses eine gute Wahl.

Entwickeln mit AWS Glue Studio

Der visuelle Editor von AWS Glue Studio ist eine grafische Oberfläche, die es einfach macht, ETL-Jobs (Extrahieren, Transformieren und Laden) in AWS Glue zu erstellen, auszuführen und zu überwachen. Sie können Workflows zur Datentransformation visuell zusammenstellen und sie nahtlos auf der Apache Spark-basierten serverlosen ETL-Engine von AWS Glue ausführen. Sie können das Schema und die Datenergebnisse in jedem Schritt des Auftrags überprüfen. Weitere Informationen finden Sie im AWS Glue Studio-Benutzerhandbuch.

Entwickeln mit interaktiven Sitzungen

Mit interaktiven Sitzungen können Sie Anwendungen aus der Umgebung Ihrer Wahl erstellen und testen. Weitere Informationen finden Sie unter Interaktive Sitzungen mit AWS Glue verwenden.

Entwickeln und testen Sie AWS Glue Jobs lokal mit einem Docker-Image

Für eine produktionsreife Datenplattform sind der Entwicklungsprozess und die CI/CD Pipeline für AWS Glue Jobs ein zentrales Thema. Sie können AWS Glue Jobs in einem Docker-Container

flexibel entwickeln und testen. AWS Glue hostet Docker-Images auf Docker Hub, um Ihre Entwicklungsumgebung mit zusätzlichen Dienstprogrammen einzurichten. Sie können Ihre bevorzugte IDE, Ihr bevorzugtes Notebook oder REPL mithilfe der ETL-Bibliothek verwenden AWS Glue . In diesem Thema wird beschrieben, wie Sie Jobs der AWS Glue Version 5.0 in einem Docker-Container mithilfe eines Docker-Images entwickeln und testen.

Verfügbare Docker-Images

Die folgenden Docker-Images sind für AWS Glue Amazon ECR verfügbar:.

- Für AWS Glue Version 5.0: `public.ecr.aws/glue/aws-glue-libs:5`
- Für AWS Glue Version 4.0: `public.ecr.aws/glue/aws-glue-libs:glue_libs_4.0.0_image_01`
- Für AWS Glue Version 3.0: `public.ecr.aws/glue/aws-glue-libs:glue_libs_3.0.0_image_01`
- Für AWS Glue Version 2.0: `public.ecr.aws/glue/aws-glue-libs:glue_libs_2.0.0_image_01`

> **Note**
>
> AWS Glue Docker-Images sind sowohl mit x86_64 als auch mit arm64 kompatibel.

In diesem Beispiel verwenden wir den Container `public.ecr.aws/glue/aws-glue-libs:5` und führen ihn auf einem lokalen Computer (Mac, Windows oder Linux) aus. Dieses Container-Image wurde für Spark-Jobs der AWS Glue Version 5.0 getestet. Das Image enthält Folgendes:

- Amazon Linux 2023
- AWS Glue ETL-Bibliothek
- Apache Spark 3.5.4
- Öffnen Sie Bibliotheken im Tabellenformat; Apache Iceberg 1.7.1, Apache Hudi 0.15.0 und Delta Lake 3.3.0
- AWS Glue Datenkatalog-Client
- Amazon Redshift Konnektor für Apache Spark
- Amazon DynamoDB Anschluss für Apache Hadoop

Um Ihren Container einzurichten, rufen Sie das Bild aus der ECR Public Gallery ab und führen Sie dann den Container aus. In diesem Thema wird gezeigt, wie Sie Ihren Container je nach Ihren Anforderungen mit den folgenden Methoden ausführen können:

- `spark-submit`
- REPL-Shell (`pyspark`)
- `pytest`
- Visual Studio-Code

Voraussetzungen

Stellen Sie vor dem Starten sicher, dass Docker installiert ist und der Docker-Daemon ausgeführt wird. Installationsanweisungen finden Sie in der Docker-Dokumentation für Mac oder Linux. Der Computer, auf dem der Docker ausgeführt wird, hostet den AWS Glue Container. Stellen Sie außerdem sicher, dass Sie über mindestens 7 GB Speicherplatz für das Image auf dem Host, auf dem der Docker ausgeführt wird, verfügen.

Weitere Informationen zu Einschränkungen bei der lokalen Entwicklung von AWS Glue Code finden Sie unter Einschränkungen für die lokale Entwicklung.

Konfiguration AWS

Um AWS API-Aufrufe aus dem Container zu aktivieren, richten Sie die AWS Anmeldeinformationen wie folgt ein. In den folgenden Abschnitten werden wir dieses AWS benannte Profil verwenden.

1. Erstellen Sie ein AWS benanntes Profil.
2. Öffnen Sie es `cmd` unter Windows oder einem Terminal Mac/Linux und führen Sie den folgenden Befehl in einem Terminal aus:

```
PROFILE_NAME="<your_profile_name>"
```

In den folgenden Abschnitten verwenden wir dieses AWS benannte Profil.

Wenn Sie Docker unter Windows ausführen, wählen Sie das Docker-Symbol (klicken Sie mit der rechten Maustaste) und wählen Sie Zu Linux-Containern wechseln, bevor Sie das Image abrufen.

Führen Sie den folgenden Befehl aus, um das Image aus ECR Public abzurufen:

```
docker pull public.ecr.aws/glue/aws-glue-libs:5
```

Führen Sie den Container aus

Sie können jetzt mit diesem Image einen Container ausführen. Sie können eine der folgenden Optionen entsprechend Ihren Anforderungen auswählen.

spark-submit

Sie können ein AWS Glue Jobskript ausführen, indem Sie den `spark-submit` Befehl auf dem Container ausführen.

1. Schreiben Sie Ihr Skript und speichern Sie es wie `sample.py` im Beispiel unten und speichern Sie es mit den folgenden Befehlen unter dem `/local_path_to_workspace/src/` Verzeichnis:

   ```
   $ WORKSPACE_LOCATION=/local_path_to_workspace
   $ SCRIPT_FILE_NAME=sample.py
   $ mkdir -p ${WORKSPACE_LOCATION}/src
   $ vim ${WORKSPACE_LOCATION}/src/${SCRIPT_FILE_NAME}
   ```

2. Diese Variablen werden im folgenden Befehl docker run verwendet. Der im Befehl spark-submit unten verwendete Beispielcode (sample.py) ist im Anhang am Ende dieses Themas enthalten.

   Führen Sie den folgenden Befehl aus, um den `spark-submit`-Befehl für den Container zum Übermitteln einer neuen Spark-Anwendung auszuführen:

   ```
   $ docker run -it --rm \
       -v ~/.aws:/home
       /hadoop/.aws \
       -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
       -e AWS_PROFILE=$PROFILE_NAME \
       --name glue5_spark_submit \
       public.ecr.aws/glue/aws-glue-libs:5 \
       spark-submit /home/hadoop/workspace/src/$SCRIPT_FILE_NAME
   ```

3. (Optional) Passen Sie die Konfiguration von `spark-submit` an Ihre Umgebung an. Sie können beispielsweise Ihre Abhängigkeiten mit der `--jars`-Konfiguration übergeben. Weitere Informationen finden Sie unter Dynamisches Laden von Spark-Eigenschaften in der Spark-Dokumentation.

REPL-Shell (Pyspark)

Sie können die REPL (`read-eval-print loops`) -Shell für die interaktive Entwicklung ausführen. Führen Sie den folgenden Befehl aus, um den PySpark Befehl auf dem Container auszuführen und die REPL-Shell zu starten:

```
$ docker run -it --rm \
    -v ~/.aws:/home/hadoop/.aws \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pyspark \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

Sie werden die folgende Ausgabe sehen:

```
Python 3.11.6 (main, Jan  9 2025, 00:00:00) [GCC 11.4.1 20230605 (Red Hat 11.4.1-2)] on
 linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
 setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.4-amzn-0
      /_/

Using Python version 3.11.6 (main, Jan  9 2025 00:00:00)
Spark context Web UI available at None
Spark context available as 'sc' (master = local[*], app id = local-1740643079929).
SparkSession available as 'spark'.
>>>
```

Mit dieser REPL-Shell können Sie interaktiv programmieren und testen.

Pytest

Für Komponententests können Sie Jobskripte `pytest` für AWS Glue Spark verwenden. Führen Sie zur Vorbereitung die folgenden Befehle aus.

```
$ WORKSPACE_LOCATION=/local_path_to_workspace
```

```
$ SCRIPT_FILE_NAME=sample.py
$ UNIT_TEST_FILE_NAME=test_sample.py
$ mkdir -p ${WORKSPACE_LOCATION}/tests
$ vim ${WORKSPACE_LOCATION}/tests/${UNIT_TEST_FILE_NAME}
```

Führen Sie den folgenden Befehl aus, um Folgendes auszuführen `pytestdocker run`:

```
$ docker run -i --rm \
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    --workdir /home/hadoop/workspace \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pytest \
    public.ecr.aws/glue/aws-glue-libs:5 \
    -c "python3 -m pytest --disable-warnings"
```

Sobald die Ausführung der Komponententests `pytest` abgeschlossen ist, sieht Ihre Ausgabe ungefähr so aus:

```
============================= test session starts ==============================
platform linux -- Python 3.11.6, pytest-8.3.4, pluggy-1.5.0
rootdir: /home/hadoop/workspace
plugins: integration-mark-0.2.0
collected 1 item

tests/test_sample.py .                                                   [100%]

======================== 1 passed, 1 warning in 34.28s =========================
```

Einrichten des Containers für die Verwendung von Visual Studio Code

Gehen Sie wie folgt vor, um den Container mit Visual Studio Code einzurichten:

1. Installieren Sie Visual Studio Code.
2. Installieren Sie Python.
3. Installieren Sie Visual Studio Code Remote - Containers
4. Öffnen Sie den Workspace-Ordner in Visual Studio Code.
5. Drücken Sie `Ctrl+Shift+P` (Windows/Linux) oder `Cmd+Shift+P` (Mac).
6. Typ `Preferences: Open Workspace Settings (JSON)`.

7. Drücken Sie die Eingabetaste.
8. Fügen Sie das folgende JSON ein und speichern Sie die Einstellung.

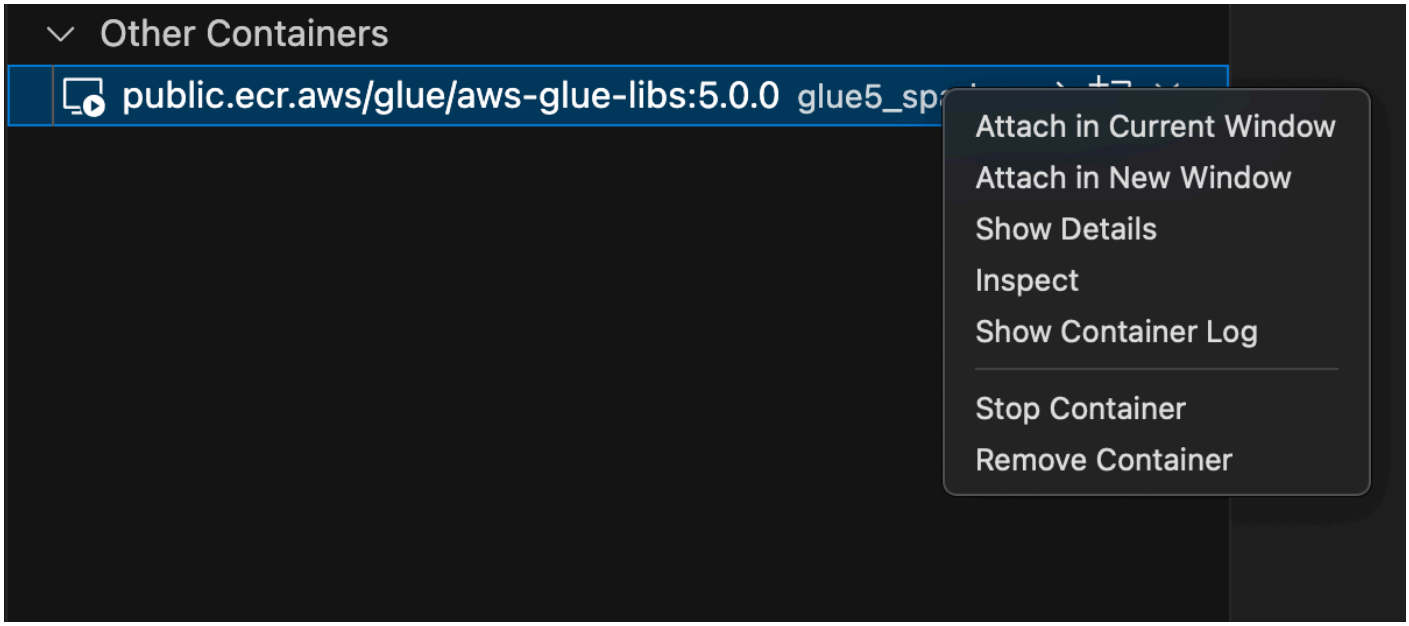
```
{
    "python.defaultInterpreterPath": "/usr/bin/python3.11",
    "python.analysis.extraPaths": [
        "/usr/lib/spark/python/lib/py4j-0.10.9.7-src.zip:/usr/lib/spark/python/:/usr/
lib/spark/python/lib/",
    ]
}
```

So richten Sie den Container ein:

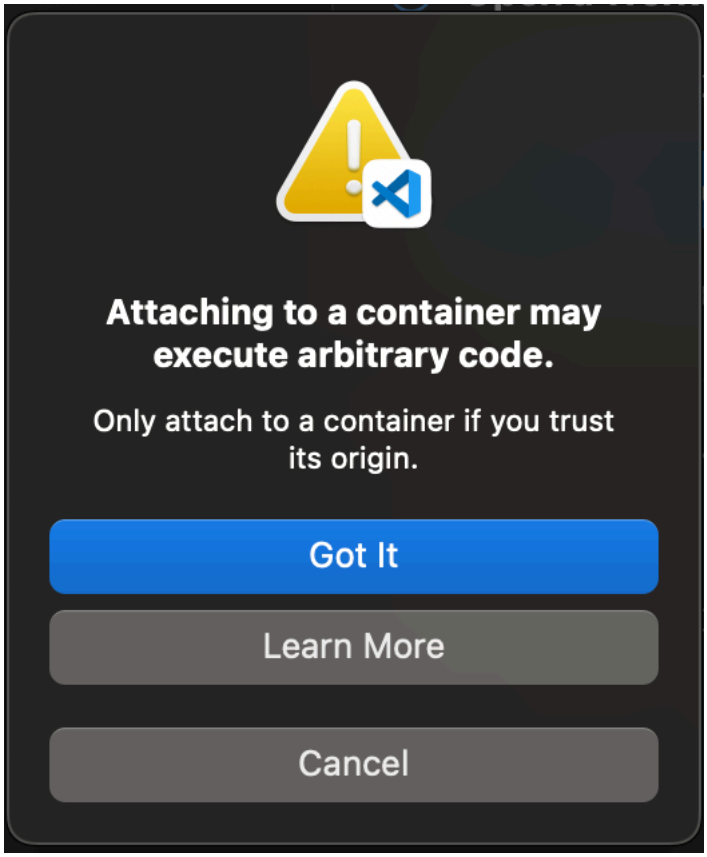
1. Führen Sie den Docker-Container aus.

```
$ docker run -it --rm \
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pyspark \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

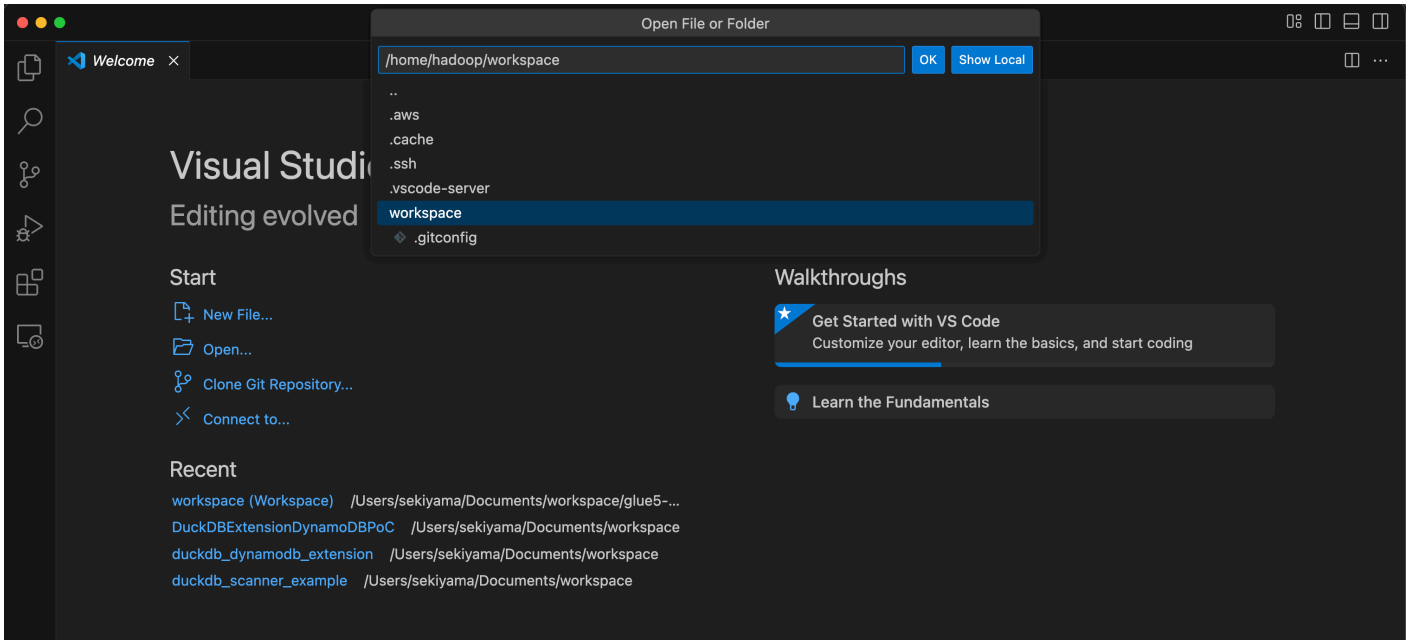
2. Starten Sie Visual Studio Code.
3. Wählen Sie im linken Menü Remote Explorer und dann `amazon/aws-glue-libs:glue_libs_4.0.0_image_01` aus.
4. Klicken Sie mit der rechten Maustaste und wählen Sie Im aktuellen Fenster anhängen.



5. Wenn das folgende Dialogfeld angezeigt wird, wählen Sie Verstanden.

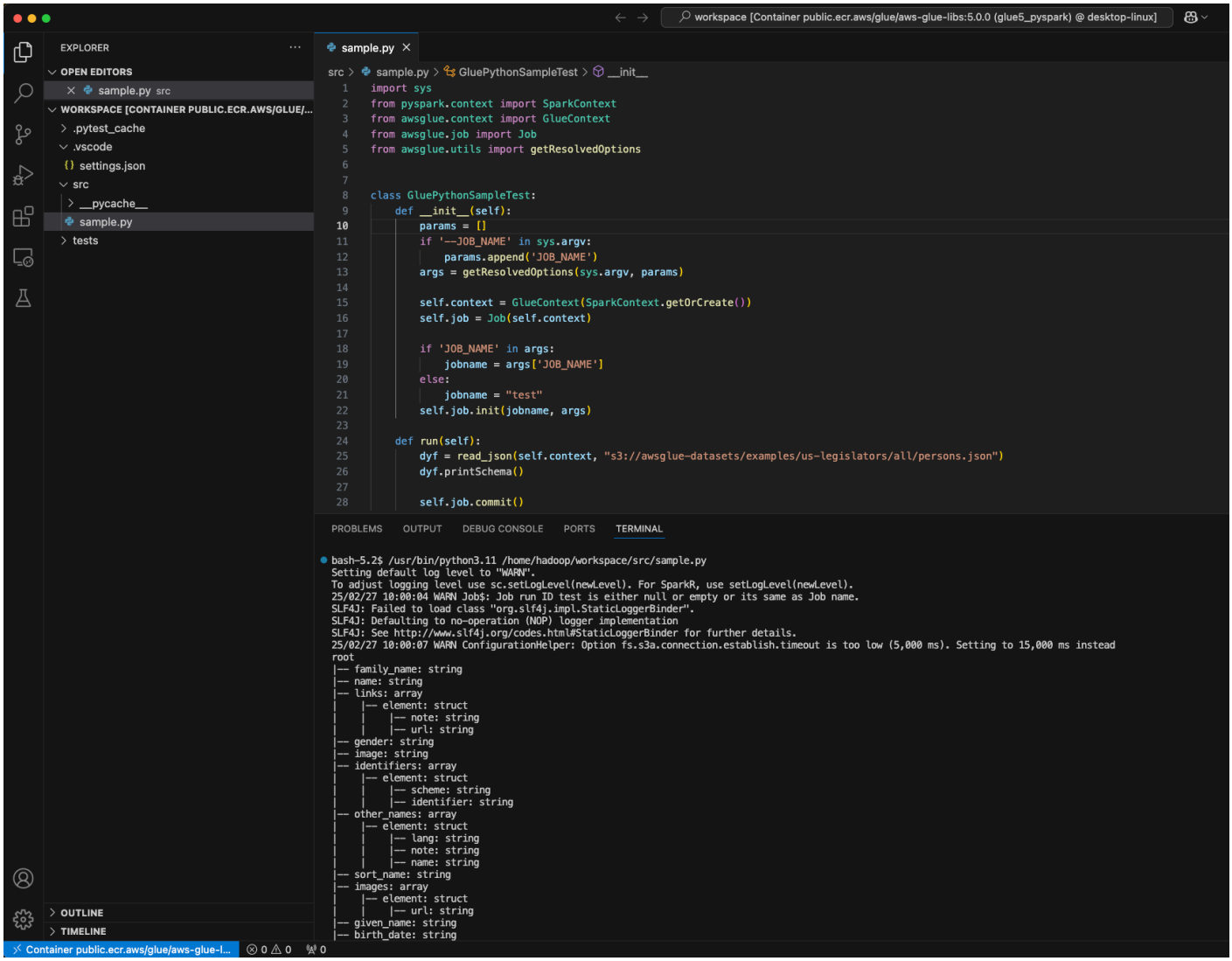


6. Öffnen Sie `/home/handoop/workspace/`.



7. Erstellen Sie ein AWS Glue PySpark Skript und wählen Sie Ausführen.

   Sie werden die erfolgreiche Ausführung des Skripts sehen.



Änderungen zwischen dem Docker-Image AWS Glue 4.0 und AWS Glue 5.0

Die wichtigsten Änderungen zwischen dem Docker-Image AWS Glue 4.0 und AWS Glue 5.0:

- In AWS Glue 5.0 gibt es ein einzelnes Container-Image für Batch- und Streaming-Jobs. Dies unterscheidet sich von Glue 4.0, wo es ein Bild für Batch und ein anderes für Streaming gab.
- In AWS Glue 5.0 lautet der Standardbenutzername des Containers`hadoop`. In AWS Glue 4.0 lautete der Standardbenutzername`glue_user`.
- In AWS Glue 5.0 wurden mehrere zusätzliche Bibliotheken, darunter Livy JupyterLab und Livy, aus dem Bild entfernt. Sie können sie manuell installieren.
- In AWS Glue 5.0 sind alle Iceberg-, Hudi- und Delta-Bibliotheken standardmäßig vorinstalliert, und die Umgebungsvariable `DATALAKE_FORMATS` wird nicht mehr benötigt. Vor AWS Glue 4.0 wurde

die Umgebungsvariable `DATALAKE_FORMATS` Umgebungsvariable verwendet, um anzugeben, welche spezifischen Tabellenformate geladen werden sollten.

Die obige Liste ist spezifisch für das Docker-Image. Weitere Informationen zu AWS Glue 5.0-Updates finden Sie unter Einführung von AWS Glue 5.0 für Apache Spark und Migration von AWS Glue for Spark-Jobs auf AWS Glue Version 5.0.

Überlegungen

Beachten Sie, dass die folgenden Funktionen nicht unterstützt werden, wenn Sie das AWS Glue Container-Image zur lokalen Entwicklung von Jobskripten verwenden.

- Auftrags-Lesezeichen
- AWS Glue Parquet Writer (unter Verwendung des Parquet-Formats in AWS Glue)
- FillMissingValues transformieren
- FindMatches transformieren
- Vektorisierter SIMD CSV-Leser
- Die Eigenschaft customJdbcDriverS3Path zum Laden des JDBC-Treibers aus dem Amazon S3-Pfad
- AWS Glue Qualität der Daten
- Erkennung sensibler Daten
- AWS Lake Formation Verkauf von Zugangsdaten auf Basis von Genehmigungen

Anhang: Hinzufügen von JDBC-Treibern und Java-Bibliotheken

Um den JDBC-Treiber hinzuzufügen, der derzeit nicht im Container verfügbar ist, können Sie unter Ihrem Workspace ein neues Verzeichnis mit den benötigten JAR-Dateien erstellen und das Verzeichnis `/opt/spark/jars/` im Befehl docker run mounten. JAR-Dateien, die sich unter dem `/opt/spark/jars/` Container befinden, werden automatisch zu Spark Classpath hinzugefügt und können während der Jobausführung verwendet werden.

Verwenden Sie beispielsweise den folgenden Befehl docker run, um JDBC-Treiber-Jars zur REPL-Shell hinzuzufügen. PySpark

```
docker run -it --rm \
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
```

```
    -v $WORKSPACE_LOCATION/jars/:/opt/spark/jars/ \
    --workdir /home/hadoop/workspace \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_jdbc \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

Wie unter Überlegungen hervorgehoben, kann die `customJdbcDriverS3Path` Verbindungsoption nicht verwendet werden, um einen benutzerdefinierten JDBC-Treiber aus Amazon S3 in AWS Glue Container-Images zu importieren.

Entwicklungsendpunkte

> **Note**
>
> Die Konsolenerfahrung für Entwicklerendpunkte wurde ab dem 31. März 2023 entfernt. Das Erstellen, Aktualisieren und Überwachen von Entwickler-Endpunkten ist weiterhin über das API für Entwicklungsendpunkte und verfügbar AWS Glue CLI.

Aus den unten aufgeführten Gründen empfehlen wir dringend, von Entwicklungsendpunkten zu interaktiven Sitzungen zu migrieren. Erforderliche Maßnahmen zum Migrieren von Entwicklungsendpunkten zu interaktiven Sitzungen finden Sie unter Migrieren von Entwicklungsendpunkten zu interaktiven Sitzungen.

Beschreibung	Entwicklungsendpunkte	Interaktive Sitzungen
Unterstützung der Glue-Version	Unterstützt AWS Glue Version 0.9 und 1.0	Unterstützt AWS Glue Version 2.0 und höher
Entwicklungsendpunkte sind in Zukunft nicht mehr in Asien-Pazifik (Jakarta) (`ap-southeast-3`), im Nahen Osten (VAE) (`me-central-1`), in Europa (Spanien) (`eu-south-2`), in Europa (Zürich) (`eu-central-2`) oder in	Interaktive Sitzungen und Notebooks sind in der Region Naher Osten (VAE) (`me-central-1`) derzeit nicht verfügbar, können aber zu einem späteren Zeitpunkt verfügbar gemacht werden	

Beschreibung	Entwicklungsendpunkte	Interaktive Sitzungen
anderen neuen Regionen verfügbar		
Zugriffsmethode auf den Spark-Cluster	Unterstützt SSH, REPL-Shell, Jupyter-Notebook, IDE (z. B.) PyCharm	unterstützt AWS Glue Studio Notizbuch, Jupyter-Notizbuch, verschiedene IDEs (z. B. Visual Studio Code PyCharm) und KI-Notizbuch SageMaker
Zeit bis zur ersten Abfrage	Erfordert 10–15 Minuten zum Einrichten eines Spark-Clusters	Das Einrichten eines kurzlebigen Spark-Clusters kann bis zu 1 Minute dauern
Preismodell	AWS Gebühren für Entwicklungsendpunkte basieren auf der Zeit, zu der der Endpunkt bereitgestellt wird, und der Anzahl der. DPUs Bei den Entwicklungsendpunkten gibt es kein Timeout. Für jeden bereitgestellten Entwicklungsendpunkt gibt es eine Mindestabrechnungsdauer von 10 Minuten. Darüber hinaus AWS fallen Gebühren für Jupyter-Notebooks auf EC2 Amazon-Instances und SageMaker KI-Notebooks an, wenn Sie sie mit Entwicklungsendpunkten konfigurieren.	AWS Gebühren für interaktive Sitzungen basieren auf der Zeit, in der die Sitzung aktiv ist, und der Anzahl der DPUs. Interaktive Sitzungen haben konfigurierbare Leerlauf-Timeouts. AWS Glue Studio Notebooks bieten eine integrierte Oberfläche für interaktive Sitzungen und werden ohne zusätzliche Kosten angeboten. Für jede interaktive Sitzung gilt eine Mindestabrechnungsdauer von 1 Minute. AWS Glue Studio Notebooks bieten eine integrierte Oberfläche für interaktive Sitzungen und werden ohne zusätzliche Kosten angeboten
Konsolenerfahrung	Nur über die CLI und API verfügbar	Erhältlich über AWS Glue Konsole, CLI und APIs

Migrieren von Entwicklungsendpunkten zu interaktiven Sitzungen

Verwenden Sie die folgende Checkliste, um die geeignete Methode zum Migrieren von Entwicklungsendpunkten zu interaktiven Sitzungen zu bestimmen.

Hängt dein Drehbuch davon ab AWS Glue 0.9- oder 1.0-spezifische Funktionen (z. B. HDFS, YARN usw.)?

Falls die Antwort Ja lautet, finden Sie weitere Informationen unter Migrieren AWS Glue Jobs zu AWS Glue Version 3.0. um zu erfahren, wie man von Glue 0.9 oder 1.0 auf Glue 3.0 und höher migriert.

Welche Methode verwenden Sie für den Zugriff auf Ihren Entwicklerendpunkt?

Wenn Sie diese Methode verwenden	Dann tun Sie das
SageMaker KI-Notizbuch, Jupyter-Notizbuch oder JupyterLab	Migrieren zu AWS Glue Studio Notizbuch , indem Sie `.ipynb` Dateien auf Jupyter herunterladen und ein neues erstellen AWS Glue Studio Notebook-Job durch Hochladen der Datei. `.ipynb` Alternativ können Sie auch SageMaker AI Studio verwenden und die AWS Glue Kernel.
Zeppelin Notebook	Konvertieren Sie das Notebook manuell in ein Jupyter Notebook, indem Sie Code kopieren und einfügen, oder automatisch mit einem Konverter eines Drittanbieters wie ze2nb. Verwenden Sie dann das Notizbuch in AWS Glue Studio Notizbuch oder SageMaker AI Studio.
IDE	Siehe Autor AWS Glue Jobs PyCharm mit AWS Glue interaktive Sitzungen oder Verwenden interaktiver Sitzungen mit Microsoft Visual Studio Code.
REPL	installieren Sie das `aws-glue-session` `package` lokal und führen Sie anschließend den folgenden Befehl aus:

Wenn Sie diese Methode verwenden	Dann tun Sie das
	• Für Python: `jupyter console --kernal glue_pyspark` • Für Scala: `jupyter console --kernal glue_spark`
SSH	Hier gibt es keine entsprechende Option für interaktive Sitzungen. Alternativ können Sie ein Docker-Image verwenden. Weitere Informati onen finden Sie unter Entwickeln mit einem Docker-Image.

In den folgenden Abschnitten erhalten Sie Informationen zur Verwendung von Entwicklungsendpunkten, um Aufträge in AWS Glue Version 1.0 zu entwickeln.

Themen

- Entwickeln von Skripts unter Verwendung von Entwicklungsendpunkten
- Verwalten von Notebooks

Entwickeln von Skripts unter Verwendung von Entwicklungsendpunkten

> **Note**
>
> Entwicklungsendpunkte werden nur für Versionen AWS Glue vor 2.0 unterstützt. Für eine interaktive Umgebung, in der Sie ETL-Skripts erstellen und testen können, verwenden Sie Notebooks on AWS Glue Studio.

AWS Glue kann eine Umgebung — einen sogenannten Entwicklungsendpunkt — erstellen, mit der Sie Ihre ETL-Skripts (Extrahieren, Transformieren und Laden) iterativ entwickeln und testen können. Sie können Entwicklungsendpunkte erstellen, bearbeiten und löschen, indem Sie AWS Glue Konsole oder API.

Verwalten Ihrer lokalen Entwicklungsumgebung

Wenn Sie einen Entwicklungsendpunkt erstellen, geben Sie Konfigurationswerte an, um die Entwicklungsumgebung bereitzustellen. Diese Werte sagen AWS Glue wie Sie das Netzwerk so einrichten, dass Sie sicher auf den Endpunkt zugreifen können und der Endpunkt auf Ihre Datenspeicher zugreifen kann.

Anschließend erstellen Sie ein Notebook, das eine Verbindung mit dem Entwicklungsendpunkt herstellt, und verwenden das Notebook zum Schreiben und Testen Ihres ETL-Skripts. Wenn Sie mit den Ergebnissen Ihres Entwicklungsprozesses zufrieden sind, können Sie einen ETL-Auftrag zum Ausführen des Skripts erstellen. Mit diesem Prozess können Sie Funktionen hinzufügen und Ihr Skript interaktive debuggen.

In den Tutorials in diesem Abschnitt erfahren Sie, wie Sie Entwicklungsendpunkte mit Notizbüchern verwenden.

Themen

- Workflow für Entwicklungsendpunkte
- Wie AWS Glue Entwicklungsendpunkte funktionieren mit Notebooks SageMaker
- Hinzufügen eines Entwicklungsendpunkts
- Zugreifen auf den Entwicklungsendpunkt
- Tutorial: Richten Sie ein Jupyter-Notebook ein, um ETL-Skripte JupyterLab zu testen und zu debuggen
- Tutorial: Verwenden Sie ein SageMaker KI-Notizbuch mit Ihrem Entwicklungsendpunkt
- Tutorial: Verwenden Sie eine REPL Shell mit Ihrem Entwicklungsendpunkt
- Tutorial: PyCharm Professional mit einem Entwicklungsendpunkt einrichten
- Erweiterte Konfiguration: Freigeben von Entwicklungsendpunkten unter mehreren Benutzern

Workflow für Entwicklungsendpunkte

Um eine zu verwenden AWS Glue Bei einem Entwicklungsendpunkt können Sie diesem Arbeitsablauf folgen:

1. Erstellen Sie mithilfe der API einen Entwicklungsendpunkt. Dieser Endpunkt wird in einer Virtual Private Cloud (VPC) mit den von Ihnen definierten Sicherheitsgruppen gestartet.

2. Die API fragt den Entwicklungsendpunkt ab, bis dieser bereitgestellt und einsatzbereit ist. Wenn er fertig ist, stellen Sie mithilfe einer der folgenden Methoden eine Verbindung zum Entwicklungsendpunkt her, um ihn zu erstellen und zu testen AWS Glue Skripte.

   - Erstellen Sie ein SageMaker KI-Notizbuch in Ihrem Konto. Weitere Informationen zum Erstellen eines Notebooks finden Sie unter the section called “Code erstellen mit AWS Glue Studio notebooks”.
   - Öffnen Sie ein Terminalfenster, um eine direkte Verbindung zu einem Entwicklungsendpunkt herzustellen.
   - Wenn Sie die Professional Edition der JetBrains PyCharm Python-IDE haben, verbinden Sie sie mit einem Entwicklungsendpunkt und verwenden Sie sie für die interaktive Entwicklung. Wenn Sie `pydevd` Anweisungen in Ihr Skript einfügen, PyCharm kann Remote-Breakpoints unterstützt werden.

3. Wenn Sie mit dem Debuggen und Testen auf Ihrem Entwicklungsendpunkt fertig sind, können Sie ihn löschen.

Wie AWS Glue Entwicklungsendpunkte funktionieren mit Notebooks SageMaker

Eine der häufigsten Möglichkeiten, auf Ihre Entwicklungsendpunkte zuzugreifen, ist die Verwendung von Jupyter auf Notebooks. SageMaker Das Jupyter Notebook ist eine Open-Source-Webanwendung, die häufig für die Visualisierung, die Analytik, Machine Learning und in ähnlichen Anwendungsfällen genutzt wird. Importieren in &S3; AWS Glue SageMaker notebook bietet Ihnen ein Jupyter-Notebook-Erlebnis mit AWS Glue Entwicklungsendpunkte. Im AWS Glue SageMaker Notebook, mit dem die Jupyter-Notebook-Umgebung vorkonfiguriert ist SparkMagic, ein Open-Source-Jupyter-Plugin zum Senden von Spark-Jobs an einen Remote-Spark-Cluster. Apache Livy ist ein Service, der die Interaktion mit einem Remote-Spark-Cluster über eine REST-API ermöglicht. Im AWS Glue SageMaker notebook, SparkMagic ist so konfiguriert, dass es die REST-API für einen Livy-Server aufruft, der auf einem AWS Glue Entwicklungsendpunkt.

Im folgenden Textfluss wird die Funktionsweise der einzelnen Komponenten erläutert:

AWS Glue SageMaker Notizbuch: (Jupyter → SparkMagic) → (Netzwerk) → AWS Glue Entwicklungsendpunkt: (Apache Livy → Apache Spark)

Sobald Sie Ihr in jedem Absatz geschriebenes Spark-Skript auf einem Jupyter-Notizbuch ausgeführt haben, wird der Spark-Code über SparkMagic an den Livy-Server übermittelt. Anschließend wird ein Spark-Job mit dem Namen „Livy-Session-N“ auf dem Spark-Cluster ausgeführt. Dieser Auftrag

wird als Livy-Sitzung bezeichnet. Der Spark-Auftrag wird ausgeführt, während die Notebook-Sitzung aktiv ist. Der Spark-Auftrag wird beendet, wenn Sie den Jupyter-Kernel im Notebook herunterfahren oder die Sitzung abgelaufen ist. Pro Notebook-Datei (mit der Endung .ipynb) wird ein Spark-Auftrag gestartet.

Sie können ein einzelnes verwenden AWS Glue Entwicklungsendpunkt mit mehreren SageMaker Notebook-Instanzen. Sie können in jeder Notebook-Instanz mehrere SageMaker Notebook-Dateien erstellen. Wenn Sie jede Notebook-Datei öffnen und die Absätze ausführen, wird eine Livy-Sitzung pro Notebook-Datei auf dem Spark-Cluster über SparkMagic gestartet. Jede Livy-Sitzung entspricht einem einzelnen Spark-Auftrag.

Standardverhalten für AWS Glue Entwicklungsendpunkte und Notebooks SageMaker

Die Spark-Aufträge werden auf der Grundlage der Spark-Konfiguration ausgeführt. Es gibt mehrere Möglichkeiten, die Spark-Konfiguration festzulegen (z. B. die Spark-Cluster-Konfiguration, SparkMagic die Konfiguration usw.).

Standardmäßig weist Spark einer Livy-Sitzung Clusterressourcen auf der Grundlage der Spark-Clusterkonfiguration zu. Im AWS Glue Entwicklungsendpunkte, die Cluster-Konfiguration hängt vom Worker-Typ ab. In dieser Tabelle werden die gängigen Konfigurationen pro Worker-Typ erläutert.

	Standard	G.1X	G.2X
`spark.driver.memory`	5G	10G	20G
`spark.executor.memory`	5G	10G	20G
`spark.executor.cores`	4	8	16
`spark.dynamicAllocation.enabled`	TRUE	TRUE	TRUE

Die maximale Anzahl von Spark-Executors wird automatisch anhand einer Kombination aus DPU (oder `NumberOfWorkers`) und Worker-Typ berechnet.

	Standard	G.1X	G.2X
Maximale Anzahl der Spark-Executors	`(DPU - 1) * 2 - 1`	`(NumberOfWorkers - 1)`	`(NumberOfWorkers - 1)`

Wenn Ihr Entwicklungsendpunkt beispielsweise 10 Worker aufweist und der Worker-Typ `G.1X` ist, haben Sie 9 Spark-Executors und der gesamte Cluster hat 90G Executor-Speicher, da jeder Executor 10G Speicher haben wird.

Unabhängig vom angegebenen Worker-Typ wird die dynamische Ressourcenzuweisung von Spark aktiviert. Wenn ein Datensatz groß genug ist, kann Spark alle Executors einer einzelnen Livy-Sitzung zuweisen, da `spark.dynamicAllocation.maxExecutors` nicht standardmäßig festgelegt ist. Das bedeutet, dass andere Livy-Sitzungen auf demselben Entwicklungsendpunkt warten, um neue Executors zu starten. Ist der Datensatz klein, können Executors mehreren Livy-Sitzungen gleichzeitig zugewiesen werden.

> **Note**
>
> Weitere Informationen dazu, wie Ressourcen in verschiedenen Anwendungsfällen zugewiesen werden und wie Sie eine Konfiguration festlegen, um das Verhalten zu ändern, finden Sie unter Erweiterte Konfiguration: Freigeben von Entwicklungsendpunkten unter mehreren Benutzern.

Hinzufügen eines Entwicklungsendpunkts

Verwenden Sie Entwicklungsendpunkte, um Ihre ETL-Skripts (Extrahieren, Transformieren und Laden) iterativ zu entwickeln und zu testen AWS Glue. Die Arbeit mit Entwicklungsendpunkten ist nur über die AWS Command Line Interface möglich.

1. Geben Sie in einem Befehlszeilenfenster einen Befehl wie den folgenden ein.

```
aws glue create-dev-endpoint --endpoint-name "endpoint1" --role-arn
 "arn:aws:iam::account-id:role/role-name" --number-of-nodes "3" --glue-version
 "1.0" --arguments '{"GLUE_PYTHON_VERSION": "3"}' --region "region-name"
```

Dieser Befehl gibt an AWS Glue Version 1.0. Da diese Version sowohl Python 2 als auch Python 3 unterstützt, können Sie den Parameter `arguments` verwenden, um die gewünschte Python-Version anzugeben. Wenn der `glue-version` Parameter weggelassen wird, AWS Glue Version 0.9 wird vorausgesetzt. Weitere Informationen zur AWS Glue Versionen finden Sie unterGlue version job property.

Informationen zu zusätzlichen Befehlszeilenparametern finden Sie create-dev-endpointin der AWS CLI Befehlsreferenz.

2. (Optional) Geben Sie den folgenden Befehl ein, um den Status des Entwicklungsendpunkts zu überprüfen. Wenn der Status in `READY` geändert wird, können Sie den Entwicklungsendpunkt verwenden.

```
aws glue get-dev-endpoint --endpoint-name "endpoint1"
```

Zugreifen auf den Entwicklungsendpunkt

Wenn Sie einen Entwicklungsendpunkt in einer Virtual Private Cloud (VPC) erstellen, AWS Glue gibt nur eine private IP-Adresse zurück. Das Feld für die öffentliche IP-Adresse wird nicht ausgefüllt. Wenn Sie einen Nicht-VPC-Entwicklungsendpunkt erstellen, AWS Glue gibt nur eine öffentliche IP-Adresse zurück.

Wenn Ihr Entwicklungsendpunkt eine Public address (Öffentliche Adresse) besitzt, überprüfen Sie, ob er mit dem privaten SSH-Schlüssel für den Entwicklungsendpunkt erreichbar ist, wie im folgenden Beispiel gezeigt.

```
ssh -i dev-endpoint-private-key.pem glue@public-address
```

Angenommen, Ihr Entwicklungsendpunkt besitzt eine Private address (Private Adresse), Ihr VPC-Subnetz ist über das öffentliche Internet erreichbar und seine Sicherheitsgruppen unterstützen den eingehenden Zugriff über Ihren Client. In diesem Fall führen Sie die folgenden Schritte aus, um einem Entwicklungsendpunkt eine elastische IP-Adresse anzufügen, sodass der Zugriff aus dem Internet unterstützt wird.

> **Note**
>
> Wenn Sie elastische IP-Adressen verwenden möchten, benötigt das verwendete Subnetz einen über die Routing-Tabelle verknüpften Internet-Gateway.

So greifen Sie durch Anfügen einer elastischen IP-Adresse auf einen Entwicklungsendpunkt zu

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Dev endpoints (Entwicklungsendpunkte) aus und navigieren Sie zur Detailseite des Entwicklungsendpunkts. Notieren Sie die Private address (Private Adresse) für den nächsten Schritt.
3. Öffnen Sie die EC2 Amazon-Konsole unter https://console.aws.amazon.com/ec2/.
4. Klicken Sie im Navigationsbereich unter Network & Security (Netzwerk und Sicherheit) auf Network Interfaces (Netzwerkschnittstellen).
5. Suchen Sie nach dem privaten DNS (IPv4), der der privaten Adresse auf der AWS Glue Seite mit den Details zum Endpunkt der Konsolenentwicklung.

   Möglicherweise müssen Sie ändern, welche Spalten auf Ihrer EC2 Amazon-Konsole angezeigt werden. Beachten Sie die Network interface ID (Netzwerkschnittstellen-ID) (ENI) für diese Adresse (z. B. `eni-12345678`).
6. Wählen Sie auf der EC2 Amazon-Konsole unter Network & Security die Option Elastic aus IPs.
7. Wählen Sie Allocate new address (Neue Adresse zuweisen) und dann Allocate (Zuweisen) aus, um eine neue elastische IP-Adresse zuzuweisen.
8. Wählen Sie auf der IPsElastic-Seite die neu zugewiesene Elastic IP aus. Wählen Sie dann Actions (Aktionen) und anschließend Associate address (Adresse zuweisen) aus.
9. Führen Sie auf der Seite Associate address (Adresse zuordnen) die folgenden Schritte aus:

   - Wählen Sie für Resource type (Ressourcentyp) Network interface (Netzwerkschnittstelle) aus.
   - Geben Sie im Feld Network interface (Netzwerkschnittstelle) die Network interface ID (Netzwerk-Schnittstellen-ID) (ENI) für die private Adresse ein.
   - Wählen Sie Associate aus.
10. Überprüfen Sie, ob die neu zugeordnete elastische IP-Adresse mit dem privaten, dem Entwicklungsendpunkt zugeordneten SSH-Schlüssel erreichbar ist, wie im folgenden Beispiel gezeigt.

```
ssh -i dev-endpoint-private-key.pem glue@elastic-ip
```

Informationen zur Verwendung eines Bastion-Hosts, um SSH-Zugriff auf die private Adresse des Entwicklungsendpunkts zu erhalten, finden Sie im AWS Sicherheits-Blogbeitrag Securely Connect to Linux Instances Running in a Private Amazon VPC.

Tutorial: Richten Sie ein Jupyter-Notebook ein, um ETL-Skripte JupyterLab zu testen und zu debuggen

In diesem Tutorial verbinden Sie ein Jupyter-Notebook, das auf Ihrem lokalen Computer JupyterLab ausgeführt wird, mit einem Entwicklungsendpunkt. Sie tun dies, damit Sie interaktiv ausführen, debuggen und testen können AWS Glue Extrahieren, Transformieren und Laden (ETL) -Skripts, bevor Sie sie bereitstellen. In dieser Anleitung wird Secure Shell (SSH) -Portweiterleitung verwendet, um Ihren lokalen Computer mit einem AWS Glue Entwicklungsendpunkt. Weitere Informationen finden Sie unter Portweiterleitung bei Wikipedia.

Schritt 1: Installation JupyterLab und Sparkmagic

Sie können die Installation mit JupyterLab `conda` oder durchführen. `pip` `conda`ist ein Open-Source-Paketverwaltungssystem und ein Umgebungsverwaltungssystem, das unter Windows, MacOS und Linux läuft. `pip`ist der Paket-Installer für Python.

Unter macOS muss Xcode installiert sein, bevor Sie SparkMagic installieren können.

1. Install JupyterLab, Sparkmagic und die zugehörigen Erweiterungen.

```
$ conda install -c conda-forge jupyterlab
$ pip install sparkmagic
$ jupyter nbextension enable --py --sys-prefix widgetsnbextension
$ jupyter labextension install @jupyter-widgets/jupyterlab-manager
```

2. Überprüfen Sie das `sparkmagic`-Verzeichnis am `Location`.

```
$ pip show sparkmagic | grep Location
Location: /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-
packages
```

3. Ändern Sie Ihr Verzeichnis in das Verzeichnis`Location`, für das zurückgegeben wurde, und installieren Sie die Kernel für Scala und. PySpark

```
$ cd /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-packages
$ jupyter-kernelspec install sparkmagic/kernels/sparkkernel
$ jupyter-kernelspec install sparkmagic/kernels/pysparkkernel
```

4. Laden Sie eine einzelne `config`-Beispieldatei herunter.

```
$ curl -o ~/.sparkmagic/config.json https://raw.githubusercontent.com/jupyter-incubator/sparkmagic/master/sparkmagic/example_config.json
```

   In dieser Konfigurationsdatei können Sie Spark-bezogene Parameter wie `driverMemory` und `executorCores` festlegen.

Schritt 2: Starten JupyterLab

Wenn Sie starten JupyterLab, wird Ihr Standard-Webbrowser automatisch geöffnet und die URL `http://localhost:8888/lab/workspaces/{workspace_name}` wird angezeigt.

```
$ jupyter lab
```

Schritt 3: Initiieren der SSH-Portweiterleitung zur Verbindung mit Ihrem Entwicklungsendpunkt

Verwenden Sie als Nächstes die lokale SSH-Portweiterleitung, um einen lokalen Port (hier8998) an das Remote-Ziel weiterzuleiten, das definiert ist durch AWS Glue (`169.254.76.1:8998`).

1. Öffnen Sie ein separates Terminalfenster, das Ihnen Zugriff auf SSH gibt. Unter Microsoft Windows können Sie die von Git for Windows bereitgestellte BASH-Shell verwenden oder Cygwin installieren.
2. Führen Sie den folgenden SSH-Befehl aus, folgendermaßen abgeändert:

   - Ersetzen Sie *`private-key-file-path`* durch einen Pfad zu der `.pem`-Datei, die den privaten Schlüssel enthält, der dem öffentlichen Schlüssel entspricht, die Sie verwendet haben, um Ihren Entwicklungsendpunkt zu erstellen.
   - Wenn Sie einen anderen Port als 8998 weiterleiten, ersetzen Sie 8998 durch die Portnummer, die Sie tatsächlich lokal verwenden. Die Adresse `169.254.76.1:8998` ist der Remote-Port und wird von Ihnen nicht geändert.

- Ersetzen Sie *dev-endpoint-public-dns* durch die öffentliche DNS-Adresse Ihres Entwicklungsendpunkts. Um diese Adresse zu finden, navigieren Sie zu Ihrem Entwicklungsendpunkt im AWS Glue Konsole, wählen Sie den Namen und kopieren Sie die öffentliche Adresse, die auf der Seite mit den Endpunktdetails aufgeführt ist.

```
ssh -i private-key-file-path -NTL 8998:169.254.76.1:8998 glue@dev-endpoint-public-
dns
```

Es wird wahrscheinlich eine Warnmeldung angezeigt, die wie folgt aussieht:

```
The authenticity of host 'ec2-xx-xxx-xxx-xx.us-west-2.compute.amazonaws.com
 (xx.xxx.xxx.xx)'
can't be established.  ECDSA key fingerprint is SHA256:4e97875Brt+1wKzRko
+JflSnp21X7aTP3BcFnHYLEts.
Are you sure you want to continue connecting (yes/no)?
```

Geben Sie das Terminalfenster ein **yes** und lassen Sie es geöffnet, während Sie es verwenden JupyterLab.

3. Überprüfen Sie, ob die SSH-Portweiterleitung mit dem Entwicklungsendpunkt korrekt funktioniert.

```
$ curl localhost:8998/sessions
{"from":0,"total":0,"sessions":[]}
```

Schritt 4: Ausführen eines einfachen Skriptfragments in einem Abschnitt des Notebooks

Jetzt JupyterLab sollte Ihr Notizbuch mit Ihrem Entwicklungsendpunkt funktionieren. Geben Sie das folgende Skriptfragment in Ihr Notebook ein und führen Sie es aus.

1. Überprüfen Sie, ob Spark erfolgreich ausgeführt wird. Der folgende Befehl weist Spark an, 1 zu berechnen und anschließend den Wert zu drucken.

```
spark.sql("select 1").show()
```

2. Prüfen Sie, ob die AWS Glue Data Catalog Integration funktioniert. Im folgenden Befehl werden die Tabellen im Data Catalog aufgelistet.

```
spark.sql("show tables").show()
```

3. Überprüfen Sie, ob es sich um ein einfaches Skriptfragment handelt, das AWS Glue Bibliotheken funktionieren.

   Das folgende Skript verwendet die `persons_json`-Tabellenmetadaten im AWS Glue Data Catalog , um einen `DynamicFrame` aus Ihren Beispieldaten zu erstellen. Es druckt dann die Elementzahlen und das Schema dieser Daten aus.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create a Glue context
glueContext = GlueContext(SparkContext.getOrCreate())

# Create a DynamicFrame using the 'persons_json' table
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")

# Print out information about *this* data
print("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Die Ausgabe des Skripts sieht folgendermaßen aus.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
```

```
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Fehlerbehebung

- Wenn sich Ihr Computer während der Installation von JupyterLab hinter einem unternehmenseigenen Proxy oder einer Firewall befindet, können HTTP- und SSL-Fehler auftreten, die auf benutzerdefinierte Sicherheitsprofile zurückzuführen sind, die von den IT-Abteilungen des Unternehmens verwaltet werden.

  Im Folgenden finden Sie ein Beispiel für einen häufig auftretenden Fehler, wenn `conda` keine Verbindung zu den eigenen Repositorys herstellen kann:

  ```
  CondaHTTPError: HTTP 000 CONNECTION FAILED for url <https://repo.anaconda.com/pkgs/
  main/win-64/current_repodata.json>
  ```

  Dies kann passieren, weil Ihr Unternehmen Verbindungen zu weit verbreiteten Repositorys in Python und JavaScript Communities blockieren kann. Weitere Informationen finden Sie auf der JupyterLab Website unter Installationsprobleme.

- Wenn bei der Verbindung mit Ihrem Entwicklungsendpunkt der Fehler connection refused (Verbindung abgelehnt) auftritt, ist Ihr Entwicklungsendpunkt möglicherweise nicht mehr auf dem neuesten Stand. Versuchen Sie, einen neuen Entwicklungsendpunkt zu erstellen und erneut eine Verbindung herzustellen.

Tutorial: Verwenden Sie ein SageMaker KI-Notizbuch mit Ihrem Entwicklungsendpunkt

In AWS Glue, können Sie einen Entwicklungsendpunkt und anschließend ein SageMaker KI-Notizbuch erstellen, um Sie bei der Entwicklung Ihrer ETL- und Machine-Learning-Skripte zu unterstützen. Ein SageMaker KI-Notebook ist eine vollständig verwaltete Recheninstanz für maschinelles Lernen, auf der die Jupyter Notebook-Anwendung ausgeführt wird.

1. Im AWS Glue Konsole, wählen Sie Dev endpoints aus, um zur Liste der Entwicklungsendpunkte zu navigieren.
2. Aktivieren Sie das Kontrollkästchen neben dem Namen eines Entwicklungsendpunkts, den Sie verwenden möchten, und wählen Sie im Menü Aktion die Option Notizbuch erstellen SageMaker aus.
3. Füllen Sie die Seite Create and configure a notebook (Notebook erstellen und konfigurieren) wie folgt aus:

   a. Geben Sie einen Namen für das Notebook ein.

   b. Überprüfen Sie unter Attach to development endpoint (An Entwicklungsendpunkt anfügen) den Entwicklungsendpunkt.

   c. Erstellen Sie eine AWS Identity and Access Management (IAM-) Rolle, oder wählen Sie sie aus.

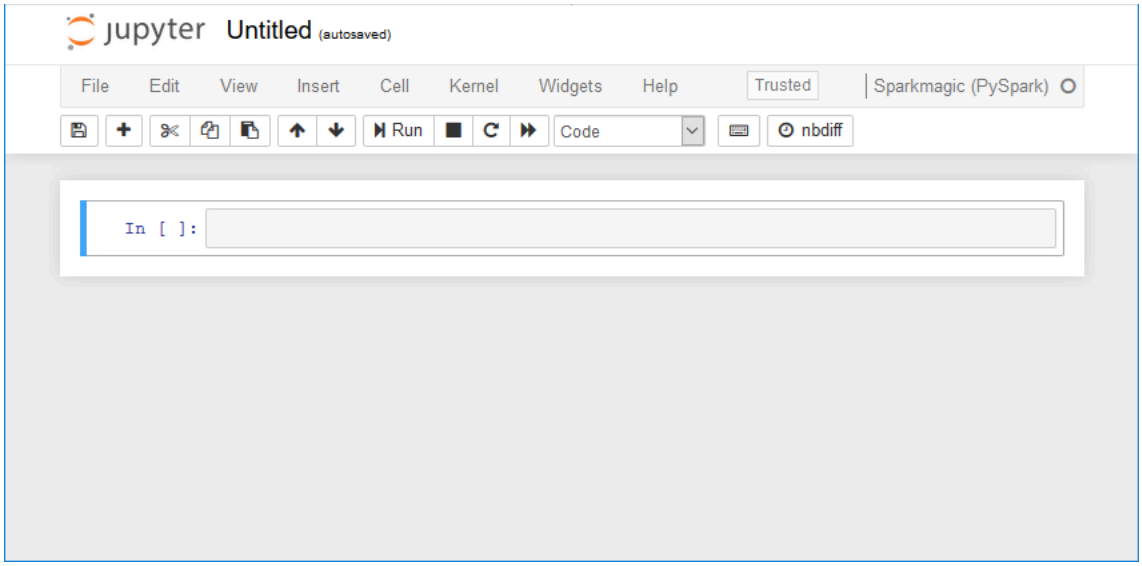
      Es wird empfohlen, eine Rolle zu erstellen. Wenn Sie eine vorhandene Rolle verwenden, stellen Sie sicher, dass sie über die erforderlichen Berechtigungen verfügt. Weitere Informationen finden Sie unter the section called “Schritt 6: Erstellen Sie eine IAM-Richtlinie für SageMaker KI-Notebooks”.

   d. (Optional) Wählen Sie eine VPC, ein Subnetz und eine oder mehrere Sicherheitsgruppen aus.

   e. (Optional) Wählen Sie einen AWS Key Management Service Verschlüsselungsschlüssel.

   f. (Optional) Fügen Sie Tags für die Notebook-Instance hinzu.
4. Klicken Sie auf Create Notebook (Notebook erstellen). Wählen Sie auf der Seite Notebooks das Aktualisierungssymbol oben rechts aus, und fahren Sie fort, bis der Status angezeigt wird `Ready`.
5. Aktivieren Sie das Kontrollkästchen neben dem neuen Notebooknamen, und wählen Sie Open notebook (Notebook öffnen).
6. Erstellen Sie ein neues Notizbuch: Wählen Sie auf der Jupyter-Seite Neu und dann Sparkmagic (). PySpark

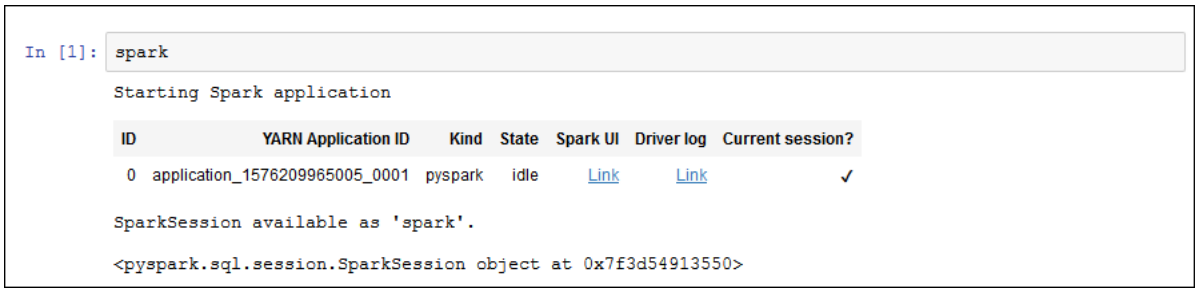
Ihr Bildschirm sollte jetzt wie folgt aussehen.



7. (Optional) Wählen Sie oben auf der Seite Untitled (Ohne Titel), und geben Sie dem Notebook einen Namen.
8. Um eine Spark-Anwendung zu starten, geben Sie den folgenden Befehl in das Notebook ein, und wählen Sie dann in der Symbolleiste Run (Ausführen) aus.

```
spark
```

Nach einer kurzen Verzögerung sollten Sie die folgende Antwort sehen:



9. Erstellen Sie einen dynamischen Frame, und führen Sie eine Abfrage aus: Kopieren, Einfügen und Ausführen des folgenden Codes, der die Anzahl und das Schema der `persons_json`-Tabelle ausgibt.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
glueContext = GlueContext(SparkContext.getOrCreate())
```

```
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Tutorial: Verwenden Sie eine REPL Shell mit Ihrem Entwicklungsendpunkt

In AWS Glue, können Sie einen Entwicklungsendpunkt erstellen und dann eine REPL-Shell (Read—Evaluate—Print Loop) aufrufen, um PySpark Code inkrementell auszuführen, sodass Sie Ihre ETL-Skripts interaktiv debuggen können, bevor Sie sie bereitstellen.

Um eine REPL auf einem Entwicklungsendpunkt verwenden zu können, benötigen Sie eine SSH-Berechtigung für den Endpunkt.

1. Öffnen Sie auf Ihrem lokalen Computer ein Terminalfenster, das SSH-Befehle ausführen kann, und fügen Sie den bearbeiteten SSH-Befehl ein. Führen Sie den Befehl aus.

   Angenommen, Sie haben zugestimmt AWS Glue Version 1.0 mit Python 3 für den Entwicklungsendpunkt, die Ausgabe wird so aussehen:

   ```
   Python 3.6.8 (default, Aug  2 2019, 17:42:44)
   [GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
   Type "help", "copyright", "credits" or "license" for more information.
   SLF4J: Class path contains multiple SLF4J bindings.
   SLF4J: Found binding in [jar:file:/usr/share/aws/glue/etl/jars/glue-assembly.jar!/
   org/slf4j/impl/StaticLoggerBinder.class]
   SLF4J: Found binding in [jar:file:/usr/lib/spark/jars/slf4j-log4j12-1.7.16.jar!/
   org/slf4j/impl/StaticLoggerBinder.class]
   SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
   SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
   Setting default log level to "WARN".
   To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
    setLogLevel(newLevel).
   2019-09-23 22:12:23,071 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
    - Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading
    libraries under SPARK_HOME.
   2019-09-23 22:12:26,562 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
    - Same name resource file:/usr/lib/spark/python/lib/pyspark.zip added multiple
    times to distributed cache
   ```

```
2019-09-23 22:12:26,580 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/etl/python/PyGlue.zip added
 multiple times to distributed cache.
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/lib/spark/python/lib/py4j-src.zip added multiple
 times to distributed cache.
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/libs/pyspark.zip added multiple
 times to distributed cache.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.3
      /_/

Using Python version 3.6.8 (default, Aug  2 2019 17:42:44)
SparkSession available as 'spark'.
>>>
```

2. Testen Sie, ob die REPL Shell ordnungsgemäß funktioniert, indem Sie die Anweisung `print(spark.version)` eingeben. Solange dies die Spark-Version anzeigt, ist Ihre REPL jetzt einsatzbereit.
3. Sie können nun versuchen, das folgende einfache Skript zeilenweise in der Shell auszuführen:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
glueContext = GlueContext(SparkContext.getOrCreate())
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Tutorial: PyCharm Professional mit einem Entwicklungsendpunkt einrichten

Dieses Tutorial zeigt Ihnen, wie Sie die PyCharm Professional Python IDE, die auf Ihrem lokalen Computer ausgeführt wird, mit einem Entwicklungsendpunkt verbinden, damit Sie sie interaktiv ausführen, debuggen und testen können. AWS Glue ETL-Skripts (Extrahieren, Übertragen und

Laden), bevor Sie sie bereitstellen. Die Anweisungen und Screenshots im Tutorial basieren auf der PyCharm Professional-Version 2019.3.

Um eine interaktive Verbindung zu einem Entwicklungsendpunkt herzustellen, muss PyCharm Professional installiert sein. Dies können Sie nicht mit der kostenlosen Version durchführen.

> **Note**
>
> Im Tutorial wird Amazon S3 als Datenquelle verwendet. Wenn Sie stattdessen eine JDBC-Datenquelle verwenden möchten, müssen Sie Ihren Entwicklungsendpunkt in einer Virtual Private Cloud (VPC) ausführen. Um eine Verbindung mit SSH zu einem Entwicklungsendpunkt in einer VPC herzustellen, müssen Sie einen SSH-Tunnel erstellen. Dieses Tutorial enthält keine Anweisungen zum Erstellen eines SSH-Tunnels. Informationen zur Verwendung von SSH für die Connect mit einem Entwicklungsendpunkt in einer VPC finden Sie unter Sichere Verbindung zu Linux-Instances, die in einer privaten Amazon-VPC ausgeführt werden, im AWS Sicherheitsblog.

Themen

- PyCharm Professionelle Verbindung zu einem Entwicklungsendpunkt herstellen
- Bereitstellen des Skripts auf Ihrem Entwicklungsendpunkt
- Konfigurieren eines Remote Interpreters
- Ausführen Ihres Skripts auf dem Entwicklungsendpunkt

PyCharm Professionelle Verbindung zu einem Entwicklungsendpunkt herstellen

1. Erstellen Sie ein neues Pure-Python-Projekt in PyCharm named. `legislators`
2. Erstellen Sie im Projekt eine Datei mit dem Namen `get_person_schema.py` und dem folgenden Inhalt:

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext


def main():
    # Create a Glue context
    glueContext = GlueContext(SparkContext.getOrCreate())
```

```
    # Create a DynamicFrame using the 'persons_json' table
    persons_DyF =
 glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")

    # Print out information about this data
    print("Count:  ", persons_DyF.count())
    persons_DyF.printSchema()


if __name__ == "__main__":
    main()
```

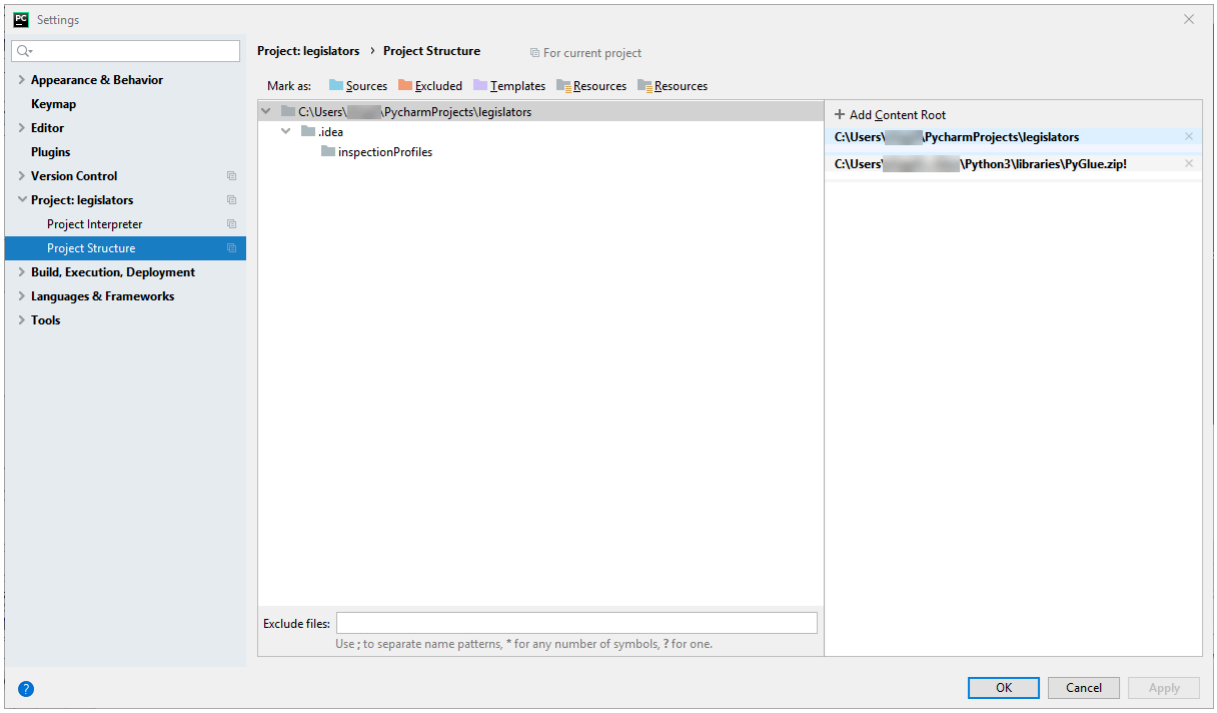
3. Führen Sie eine der folgenden Aktionen aus:

   - Wählen Sie in der &Snowconsole; Ihren Auftrag aus der Tabelle. AWS Glue Version 0.9, laden Sie die herunter AWS Glue Python-BibliotheksdateiPyGlue.zip,, von `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl/python/PyGlue.zip` zu einem geeigneten Speicherort auf Ihrem lokalen Computer.
   - Wählen Sie in der &Snowconsole; Ihren Auftrag aus der Tabelle. AWS Glue Version 1.0 und höher, laden Sie die herunter AWS Glue Python-BibliotheksdateiPyGlue.zip,, von `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl-1.0/python/PyGlue.zip` zu einem geeigneten Speicherort auf Ihrem lokalen Computer.

4. Fügen Sie `PyGlue.zip` es als Inhaltsstamm für Ihr Projekt hinzu in PyCharm:

   - Wählen Sie in PyCharm Datei, Einstellungen, um das Dialogfeld Einstellungen zu öffnen. (Sie können auch `Ctrl+Alt+S` betätigen.)
   - Erweitern Sie das Projekt `legislators` und wählen Sie Project Structure (Projektstruktur) aus. Wählen Sie anschließend auf der rechten Seite + Add Content Root (+ Inhalts-Stammverzeichnis hinzufügen) aus.
   - Navigieren Sie zum Speicherort, an dem Sie `PyGlue.zip` gespeichert haben, und wählen Sie dann Apply (Anwenden) aus.

   Der Bildschirm Settings (Einstellungen) sollte etwa wie folgt aussehen:



Lassen Sie das Dialogfeld Settings (Einstellungen) geöffnet, nachdem Sie Apply (Anwenden) ausgewählt haben.

5. Konfigurieren Sie die Bereitstellungsoptionen, um das lokale Skript mithilfe von SFTP auf Ihren Entwicklungsendpunkt hochzuladen (diese Funktion ist nur in PyCharm Professional verfügbar):

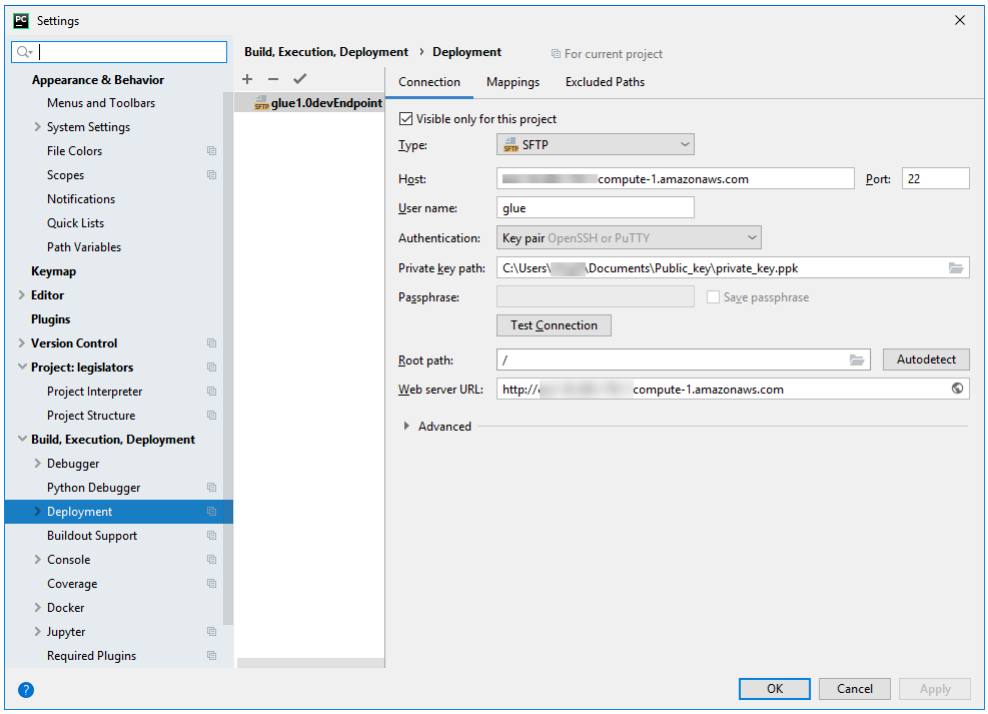
   - Erweitern Sie im Dialogfeld Settings (Einstellungen) den Abschnitt Build, Execution, Deployment (Erstellung, Ausführung, Bereitstellung). Wählen Sie den Teilabschnitt Deployment (Bereitstellung) aus.
   - Wählen Sie das Symbol + oben im mittleren Bereich aus, um einen neuen Server hinzuzufügen. Setzen Sie seinen Type (Typ) auf `SFTP`, und geben Sie ihm einen Namen.
   - Legen Sie den SFTP-Host auf die Public Address (Öffentliche Adresse) Ihres Entwicklungsendpunkts fest, wie auf der Detailseite aufgeführt. (Wählen Sie den Namen Ihres Entwicklungsendpunkts in der AWS Glue Konsole, um die Detailseite anzuzeigen). Legen Sie für einen Entwicklungsendpunkt, der in einer VPC ausgeführt wird, den SFTP-Host auf die Hostadresse und den lokalen Port des SSH-Tunnels auf den Entwicklungsendpunkt fest.
   - Legen Sie den User name (Benutzername) auf `glue` fest.
   - Legen Sie den Auth type (Authentifizierungstyp) auf Key pair (OpenSSH or Putty) (Schlüsselpaar (OpenSSH oder PuTTY)) fest. Legen Sie die Private key file (Datei für den privaten Schlüssel) fest, indem Sie zum Speicherort navigieren, an dem die Datei für den privaten Schlüssel Ihres Entwicklungsendpunkts gespeichert ist. Beachten Sie, dass PyCharm

nur die OpenSSH-Schlüsseltypen DSA, RSA und ECDSA unterstützt werden und keine Schlüssel im privaten Format von Putty akzeptiert. Sie können eine up-to-date Version von verwenden, um einen Schlüsselpaartyp `ssh-keygen` zu generieren, der akzeptiert, und zwar mit einer Syntax wie der folgenden: PyCharm

```
ssh-keygen -t rsa -f <key_file_name> -C "<your_email_address>"
```

- Wählen Sie Test SFTP connection (SFTP-Verbindung testen) aus, und lassen Sie das Testen der Verbindung zu. Wenn die Verbindung erfolgreich hergestellt wird, wählen Sie Apply (Anwenden) aus.

Der Bildschirm Settings (Einstellungen) sollte nun etwa wie folgt aussehen:



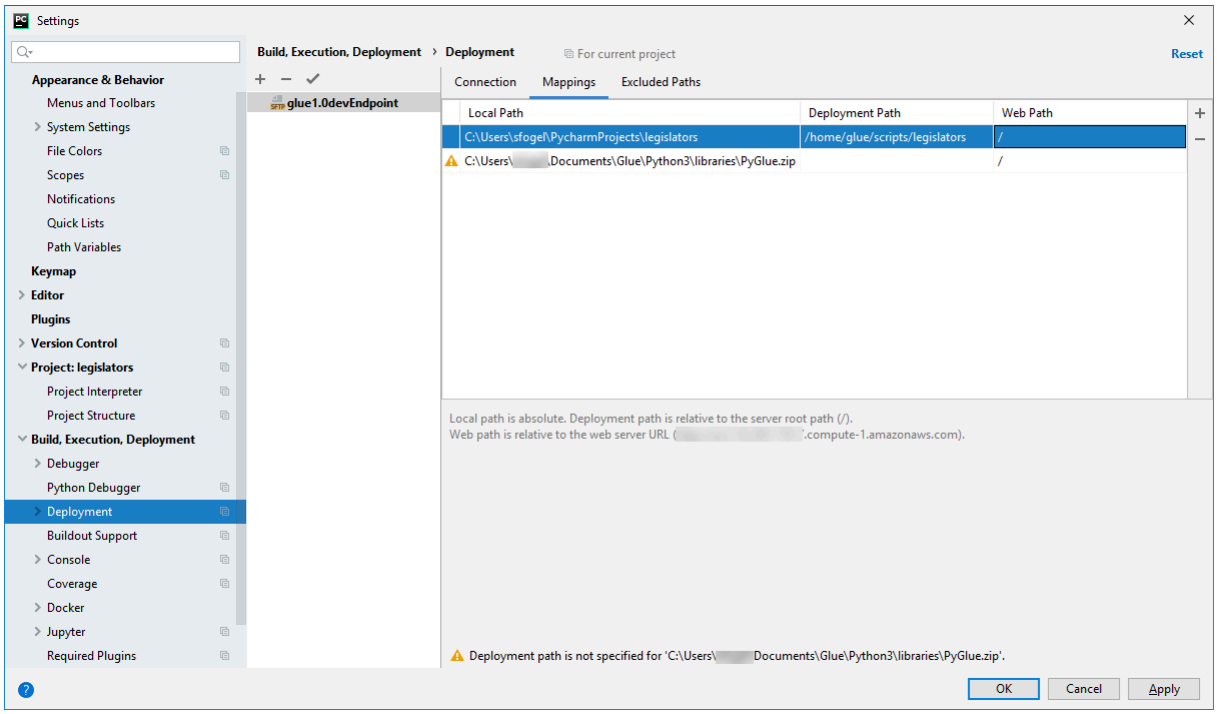
Lassen Sie wiederum das Dialogfeld Settings (Einstellungen) geöffnet, nachdem Sie Apply (Anwenden) ausgewählt haben.

6. Ordnen Sie das lokale Verzeichnis einem Remote-Verzeichnis für die Bereitstellung zu:

   - Wählen Sie rechts auf der Seite Deployment (Bereitstellung) die mittlere Registerkarte mit der Bezeichnung Mappings aus.
   - Geben Sie in der Spalte Deployment Path (Bereitstellungspfad) einen Pfad für die Bereitstellung Ihres Projekts unter `/home/glue/scripts/` ein. Beispiel: `/home/glue/scripts/legislators`.

- Wählen Sie Anwenden aus.

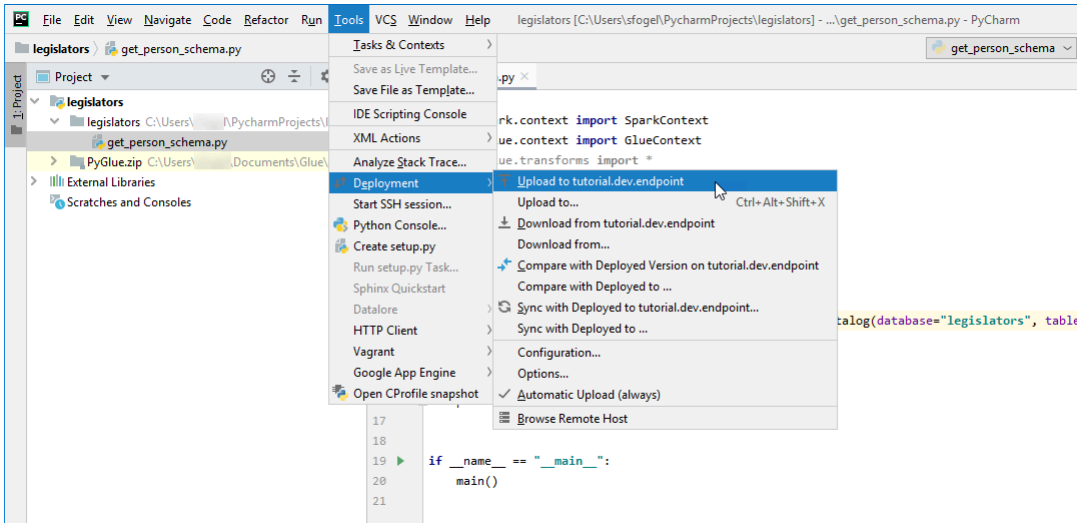
Der Bildschirm Settings (Einstellungen) sollte nun etwa wie folgt aussehen:



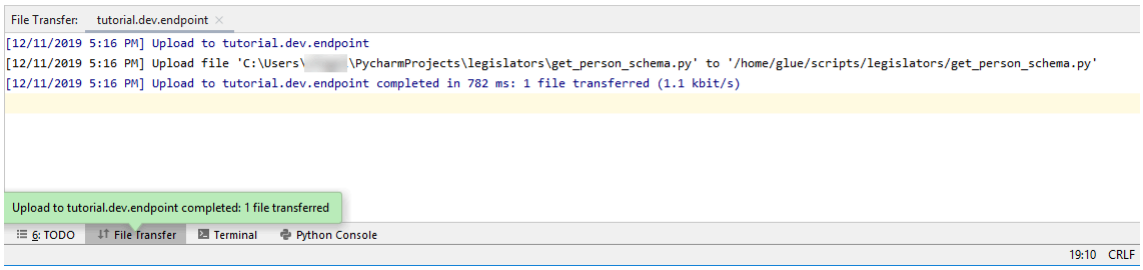
Klicken Sie auf OK, um das Dialogfeld Settings (Einstellungen) zu schließen.

Bereitstellen des Skripts auf Ihrem Entwicklungsendpunkt

1. Wählen Sie Tools (Extras), Deployment (Bereitstellung), und wählen Sie dann den Namen, unter dem Sie Ihren Entwicklungsendpunkt einrichten, wie im folgenden Image dargestellt:



Nachdem Ihr Skript bereitgestellt wurde, sollte der untere Bereich des Bildschirms etwa wie folgt aussehen:



2. Wählen Sie in der Menüleiste Tools (Extras), Deployment (Bereitstellung), Automatic Upload (always) (Automatischer Upload (immer)). Stellen Sie sicher, dass neben Automatic Upload (always) (Automatischer Upload (immer)) ein Häkchen angezeigt wird.

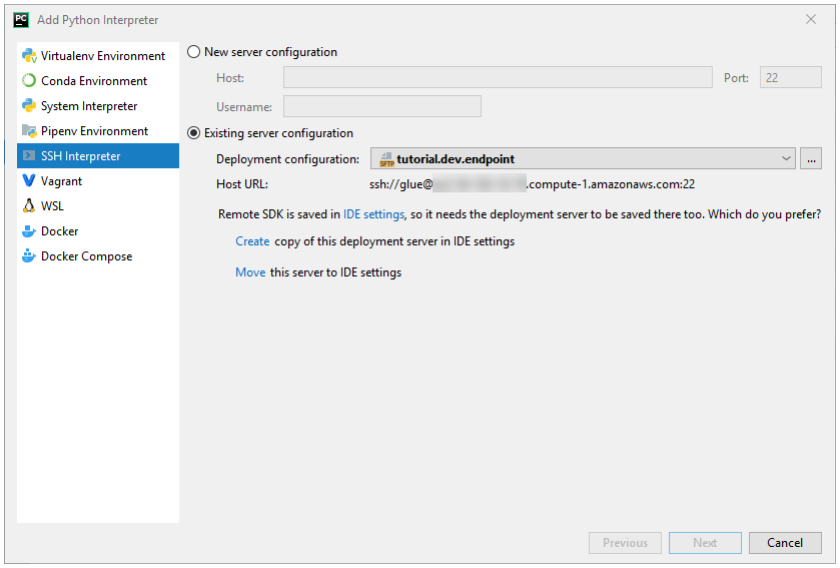
   Wenn diese Option aktiviert ist, PyCharm werden geänderte Dateien automatisch auf den Entwicklungsendpunkt hochgeladen.

Konfigurieren eines Remote Interpreters

Konfigurieren Sie PyCharm für die Verwendung des Python-Interpreters auf dem Entwicklungsendpunkt.

1. Wählen Sie im Menü File (Datei) die Option Open (Öffnen) aus.
2. Erweitern Sie die Project legislators (Projektgesetzgeber), und wählen Sie Project Interpreter.
3. Wählen Sie das Zahnradsymbol neben der Liste Project Interpreter, und wählen Sie dann Add (Hinzufügen).
4. Wählen Sie im Dialogfeld Add Python Interpreter (Python-Interpreter hinzufügen) im linken Bereich die Option SSH-Interpreter.
5. Wählen Sie Existing server configuration (Vorhandene Serverkonfiguration), und wählen Sie in der Liste Deployment configuration (Bereitstellungskonfiguration) Ihre Konfiguration aus.

   Ihr Bildschirm sollte ungefähr wie das folgende Image aussehen.



6. Wählen Sie Move this server to IDE settings (Diesen Server in IDE-Einstellungen verschieben) und dann Next (Weiter).
7. Ändern Sie im Feld Interpreter den Pfad zu `/usr/bin/gluepython`, wenn Sie Python 2 verwenden, oder zu `/usr/bin/gluepython3`, wenn Sie Python 3 verwenden. Klicken Sie auf Finish (Fertig stellen).

Ausführen Ihres Skripts auf dem Entwicklungsendpunkt

So führen Sie das Skript aus:

- Klicken Sie im linken Bereich mit der rechten Maustaste auf den Dateinamen und wählen Sie Ausführen '***<filename>***' aus.

  Nach einer Reihe von Nachrichten sollte die endgültige Ausgabe die Anzahl und das Schema anzeigen.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
```

```
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string


Process finished with exit code 0
```

Nun können Sie Ihr Skript remote auf Ihrem Entwicklungsendpunkt debuggen.

Erweiterte Konfiguration: Freigeben von Entwicklungsendpunkten unter mehreren Benutzern

In diesem Abschnitt wird erklärt, wie Sie in typischen Anwendungsfällen Entwicklungsendpunkte mit SageMaker Notebooks nutzen können, um Entwicklungsendpunkte für mehrere Benutzer gemeinsam zu nutzen.

Single-Tenancy-Konfiguration

Um die Entwicklererfahrung zu vereinfachen und Ressourcenkonflikte zu vermeiden, wird in Single-Tenant-Anwendungsfällen empfohlen, dass jeder Entwickler einen eigenen Entwicklungsendpunkt verwendet, der auf sein jeweiliges Projekt ausgerichtet ist. Dies vereinfacht auch die Entscheidungen zu Worker-Typ und DPU-Anzahl, da sie je nach Projekt und Ermessen des Entwicklers eingestellt werden können.

Solange Sie nicht mehrere Notebook-Dateien gleichzeitig ausführen, müssen Sie sich nicht um die Ressourcenzuweisung kümmern. Wenn Sie Code in mehreren Notebook-Dateien gleichzeitig ausführen, werden mehrere Livy-Sitzungen parallel gestartet. Um Spark-Cluster-Konfigurationen zu trennen, sodass mehrere Livy-Sitzungen gleichzeitig ausgeführt werden können, führen Sie die Schritte aus, die in Multi-Tenant-Anwendungsfällen eingeführt werden.

Wenn Ihr Entwicklungsendpunkt beispielsweise 10 Worker aufweist und der Worker-Typ `G.1X` ist, haben Sie 9 Spark-Executors und der gesamte Cluster hat 90G Executor-Speicher, da jeder Executor 10G Speicher haben wird.

Unabhängig vom angegebenen Worker-Typ wird die dynamische Ressourcenzuweisung von Spark aktiviert. Wenn ein Datensatz groß genug ist, kann Spark alle Executors einer einzelnen Livy-Sitzung zuweisen, da `spark.dynamicAllocation.maxExecutors` nicht standardmäßig festgelegt ist. Das bedeutet, dass andere Livy-Sitzungen auf demselben Entwicklungsendpunkt warten, um neue Executors zu starten. Ist der Datensatz klein, können Executors mehreren Livy-Sitzungen gleichzeitig zugewiesen werden.

> **Note**
>
> Weitere Informationen dazu, wie Ressourcen in verschiedenen Anwendungsfällen zugewiesen werden und wie Sie eine Konfiguration festlegen, um das Verhalten zu ändern, finden Sie unter Erweiterte Konfiguration: Freigeben von Entwicklungsendpunkten unter mehreren Benutzern.

Multi-Tenancy-Konfiguration

> **Note**
>
> Bitte beachten Sie, dass Entwicklungsendpunkte das emulieren sollen AWS Glue ETL-Umgebung als Single-Tenant-Umgebung. Multi-Tenancy ist zwar möglich, ist jedoch ein erweiterter Anwendungsfall. In der Regel wird empfohlen, dass die Benutzer für jeden Entwicklungsendpunkt ein Single-Tenancy-Muster beibehalten.

In Multi-Tenant-Anwendungsfällen müssen Sie sich gegebenenfalls um die Ressourcenzuweisung kümmern. Der entscheidende Faktor ist die Anzahl der Benutzer, die ein Jupyter Notebook gleichzeitig verwenden. Wenn Ihr Team in einem "follow-the-sun" Workflow arbeitet und es in jeder

Zeitzone nur einen Jupyter-Benutzer gibt, dann ist die Anzahl der gleichzeitigen Benutzer nur eins, sodass Sie sich keine Gedanken über die Ressourcenzuweisung machen müssen. Wird Ihr Notebook jedoch von mehreren Benutzern gemeinsam genutzt und jeder Benutzer reicht Code auf einer Ad-hoc-Basis ein, müssen Sie die folgenden Punkte berücksichtigen.

Um Spark-Cluster-Ressourcen auf mehrere Benutzer aufzuteilen, können Sie Konfigurationen verwenden. SparkMagic Es gibt zwei verschiedene Arten der Konfiguration SparkMagic.

(A) Verwenden der Anweisung %%configure -f

Wenn Sie die Konfiguration pro Livy-Sitzung des Notebooks ändern möchten, können Sie die Anweisung `%%configure -f` für den Notebook-Abschnitt ausführen.

Soll beispielsweise eine Spark-Anwendung auf 5 Executors ausgeführt werden, können Sie für den Notebook-Abschnitt den folgenden Befehl ausführen.

```
%%configure -f
{"numExecutors":5}
```

Sie sehen dann nur 5 Executors, die für den Auftrag in der Spark-Benutzeroberfläche ausgeführt werden.

Wir empfehlen, die maximale Anzahl von Executors für die dynamische Ressourcenzuweisung zu begrenzen.

```
%%configure -f
{"conf":{"spark.dynamicAllocation.maxExecutors":"5"}}
```

(B) Ändern Sie die SparkMagic Konfigurationsdatei

SparkMagic funktioniert basierend auf der Livy-API. SparkMagic erstellt Livy-Sitzungen mit Konfigurationen wie`driverMemory`, `driverCores`,`executorMemory`,`executorCores`, `numExecutorsconf`, usw. Dies sind die Schlüsselfaktoren, die bestimmen, wie viele Ressourcen vom gesamten Spark-Cluster verbraucht werden. SparkMagic ermöglicht es Ihnen, eine Konfigurationsdatei bereitzustellen, um die Parameter anzugeben, die an Livy gesendet werden. In diesem GitHub-Repository können Sie sich eine Beispielkonfigurationsdatei ansehen.

Wenn Sie die Konfiguration für alle Livy-Sitzungen eines Notebooks ändern möchten, können Sie `/home/ec2-user/.sparkmagic/config.json` anpassen, indem Sie `session_config` hinzufügen.

Um die Konfigurationsdatei auf einer SageMaker Notebook-Instanz zu ändern, können Sie die folgenden Schritte ausführen.

1. Öffnen Sie ein SageMaker Notizbuch.
2. Öffnen Sie den Terminalkernel.
3. Führen Sie die folgenden Befehle aus:

```
sh-4.2$ cd .sparkmagic
sh-4.2$ ls
config.json logs
sh-4.2$ sudo vim config.json
```

Sie können `/home/ec2-user/.sparkmagic/config.json` beispielsweise diese Zeilen hinzufügen und den Jupyter-Kernel des Notebooks neu starten.

```
  "session_configs": {
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Leitlinien und Bewährte Methoden

Um diesen Ressourcenkonflikt zu vermeiden, können Sie unter anderem diese grundlegenden Ansätze nutzen:

- Vergrößern Sie den Spark-Cluster, indem Sie die `NumberOfWorkers` erhöhen (horizontale Skalierung) und den `workerType` upgraden (vertikale Skalierung).
- Weisen Sie weniger Ressourcen pro Benutzer zu (weniger Ressourcen pro Livy-Sitzung).

Ihr Ansatz hängt vom Anwendungsfall ab. Wenn Sie einen größeren Entwicklungsendpunkt haben und keine großen Datenmengen vorliegen, ist das Risiko eines Ressourcenkonflikts wesentlich geringer, da Spark Ressourcen basierend auf einer dynamischen Zuweisungsstrategie zuteilen kann.

Die Anzahl der Spark-Executors kann wie oben beschrieben anhand einer Kombination aus DPU (oder `NumberOfWorkers`) und Worker-Typ automatisch berechnet werden. Jede Spark-Anwendung startet einen Treiber und mehrere Executors. Zur Berechnung muss Folgendes gegeben sein: `NumberOfWorkers` = `NumberOfExecutors + 1`. In der folgenden Matrix wird erläutert, wie viel

Kapazität Sie basierend auf der Anzahl der gleichzeitigen Benutzer in Ihrem Entwicklungsendpunkt benötigen.

Anzahl gleichzeitiger Notebook-Benutzer	Anzahl der Spark-Executors, die pro Benutzer zugewiesen werden sollen	Summe NumberOfWorkers für Ihren Entwickler-Endpunkt
3	5	18
10	5	60
50	5	300

Wenn Sie weniger Ressourcen pro Benutzer zuweisen möchten, wäre `spark.dynamicAllocation.maxExecutors` (oder `numExecutors`) der einfachste Parameter zur Konfiguration als Livy-Sitzungsparameter. Wenn Sie die folgende Konfiguration einstellen/`home/ec2-user/.sparkmagic/config.json`, SparkMagic werden pro Livy-Sitzung maximal 5 Executoren zugewiesen. Das hilft Ihnen, die Ressourcen pro Livy-Sitzung zu trennen.

```
"session_configs": {
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Angenommen, Sie haben einen Entwicklungsendpunkt mit 18 Workern (G.1X) und es gibt 3 gleichzeitige Notebook-Benutzer. Wenn Ihre Sitzungskonfiguration über den Parameter `spark.dynamicAllocation.maxExecutors=5` verfügt, kann jeder Benutzer 1 Treiber und 5 Executors nutzen. Selbst wenn Sie mehrere Notebook-Abschnitte gleichzeitig ausführen, treten keine Ressourcenkonflikte auf.

Nachteile

Mit dieser Sitzungskonfiguration (`"spark.dynamicAllocation.maxExecutors":"5"`) können Sie Fehler aufgrund von Ressourcenkonflikten vermeiden und müssen nicht auf die Ressourcenzuweisung warten, wenn mehrere Benutzerzugriffe gleichzeitig erfolgen. Allerdings kann Spark niemals mehr als 5 Executors für Ihre Livy-Sitzung zuweisen, selbst wenn viele freie Ressourcen verfügbar sind (z. B. weile keine anderen gleichzeitigen Benutzer vorliegen).

Sonstige Hinweise

Es wird empfohlen, den Jupyter-Kernel zu beenden, wenn Sie ein Notebook nicht mehr verwenden. Dadurch werden Ressourcen freigegeben, die andere Notebook-Benutzer sofort nutzen können, ohne auf den Ablauf des Kernels (automatisches Herunterfahren) warten zu müssen.

Häufige Probleme

Auch wenn Sie die Richtlinien befolgen, können bestimmte Probleme auftreten.

Sitzung nicht gefunden

Wenn Sie versuchen, einen Notebook-Abschnitt auszuführen, obwohl Ihre Livy-Sitzung bereits beendet wurde, wird die folgende Meldung angezeigt. Um die Livy-Sitzung zu aktivieren, müssen Sie den Jupyter-Kernel neu starten. Dazu gehen Sie im Jupyter-Menü auf Kernel > Restart (Neustart) und führen anschließend den Notebook-Abschnitt erneut aus.

```
An error was encountered:
Invalid status code '404' from http://localhost:8998/sessions/13 with error payload:
 "Session '13' not found."
```

Nicht genügend YARN-Ressourcen

Wenn Sie versuchen, einen Notebook-Abschnitt auszuführen, obwohl Ihr Spark-Cluster nicht über genügend Ressourcen verfügt, um eine neue Livy-Sitzung zu starten, wird die folgende Meldung angezeigt. Oft können Sie dieses Problem vermeiden, wenn Sie die Richtlinien einhalten. Trotzdem besteht die Möglichkeit, dass dieses Problem auftritt. Als Workaround können Sie überprüfen, ob nicht benötigte aktive Livy-Sitzungen vorhanden sind. Diese sollten beendet werden, um die Clusterressourcen freizugeben. Details finden Sie im nächsten Abschnitt.

```
Warning: The Spark session does not have enough YARN resources to start.
The code failed because of a fatal error:
    Session 16 did not start up in 60 seconds..

Some things to try:
a) Make sure Spark has enough available resources for Jupyter to create a Spark
 context.
b) Contact your Jupyter administrator to make sure the Spark magics library is
 configured correctly.
c) Restart the kernel.
```

Überwachung und Debugging

In diesem Abschnitt werden Methoden für die Überwachung von Ressourcen und Sitzungen beschrieben.

Überwachen und Debuggen der Zuweisung von Clusterressourcen

Sie können die Spark-Benutzeroberfläche beobachten, um zu überwachen, wie viele Ressourcen pro Livy-Sitzung zugewiesen werden und welche Spark-Konfigurationen für den Auftrag am effektivsten sind. Informationen zum Aktivieren der Spark-Benutzeroberfläche finden Sie unter Aktivieren der Apache-Spark-Webbenutzeroberfläche für Entwicklungsendpunkte.

(Optional) Wenn Sie eine Echtzeitansicht der Spark-Benutzeroberfläche benötigen, können Sie einen SSH-Tunnel für den Spark-Verlaufsserver konfigurieren, der auf dem Spark-Cluster ausgeführt wird.

```
ssh -i <private-key.pem> -N -L 8157:<development endpoint public address>:18080
 glue@<development endpoint public address>
```

Dann können Sie http://localhost:8157 in Ihrem Browser öffnen, um die Spark-Benutzeroberfläche aufzurufen.

Kostenlose, nicht benötigte Livy-Sitzungen

Sehen Sie sich diese Verfahren an, mit denen nicht benötigte Livy-Sitzungen in einem Notebook oder Spark-Cluster beendet werden.

(a). Beenden von Livy-Sitzungen in einem Notebook

Sie können den Kernel in einem Jupyter Notebook herunterfahren, um nicht benötigte Livy-Sitzungen zu beenden.

(b). Beenden von Livy-Sitzungen in einem Spark-Cluster

Wenn noch nicht benötigte Livy-Sitzungen ausgeführt werden, können Sie diese im Spark-Cluster beenden.

Als Voraussetzung für dieses Verfahren müssen Sie den öffentlichen SSH-Schlüssel für Ihren Entwicklungsendpunkt konfigurieren.

Zur Anmeldung beim Spark-Cluster können Sie den folgenden Befehl ausführen:

```
$ ssh -i <private-key.pem> glue@<development endpoint public address>
```

Mit dem folgenden Befehl können Sie die aktiven Livy-Sitzungen anzeigen:

```
$ yarn application -list
20/09/25 06:22:21 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/172.38.106.206:8032
Total number of applications (application-types: [] and states: [SUBMITTED, ACCEPTED,
 RUNNING]):2
Application-Id Application-Name Application-Type User Queue State Final-State Progress
 Tracking-URL
application_1601003432160_0005 livy-session-4 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-4-130.ec2.internal:41867
application_1601003432160_0004 livy-session-3 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-179-185.ec2.internal:33727
```

Anschließend können Sie die Livy-Sitzungen mit dem folgenden Befehl beenden:

```
$ yarn application -kill application_1601003432160_0005
20/09/25 06:23:38 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/255.1.106.206:8032
Killing application application_1601003432160_0005
20/09/25 06:23:39 INFO impl.YarnClientImpl: Killed application
 application_1601003432160_0005
```

Verwalten von Notebooks

> **Note**
>
> Entwicklungsendpunkte werden nur für Versionen AWS Glue vor 2.0 unterstützt. Für eine interaktive Umgebung, in der Sie ETL-Skripts erstellen und testen können, verwenden Sie Notebooks on AWS Glue Studio.

Ein Notebook ermöglicht die interaktive Entwicklung und das Testen Ihrer ETL-Skripte (Extrahieren, Transformieren und Laden) auf einem Entwicklungsendpunkt. AWS Glue bietet eine Schnittstelle zu SageMaker AI Jupyter-Notebooks. Mit AWS Glue, du erstellst und verwaltest SageMaker KI-Notizbücher. Sie können SageMaker KI-Notizbücher auch über das öffnen AWS Glue console.

Darüber hinaus können Sie Apache Spark mit SageMaker KI auf verwenden AWS Glue Entwicklungsendpunkte, die SageMaker KI unterstützen (aber nicht AWS Glue ETL-Jobs).

SageMaker Spark ist eine Open-Source-Apache Spark-Bibliothek für SageMaker KI. Weitere Informationen finden Sie unter Verwenden von Apache Spark mit Amazon SageMaker.

Important

Verwaltung von SageMaker KI-Notebooks mit AWS Glue Development Endpoints ist in den folgenden AWS Regionen verfügbar:

Region	Code
US East (Ohio)	`us-east-2`
USA Ost (Nord-Virginia)	`us-east-1`
USA West (Nordkalifornien)	`us-west-1`
USA West (Oregon)	`us-west-2`
Asien-Pazifik (Tokio)	`ap-northeast-1`
Asien-Pazifik (Seoul)	`ap-northeast-2`
Asia Pacific (Mumbai)	`ap-south-1`
Asien-Pazifik (Singapur)	`ap-southeast-1`
Asien-Pazifik (Sydney)	`ap-southeast-2`
Kanada (Zentral)	`ca-central-1`
Europe (Frankfurt)	`eu-central-1`
Europa (Irland)	`eu-west-1`
Europa (London)	`eu-west-2`

Erstellung visueller ETL-Jobs

Erstellen Sie visuelle ETL-Jobs mit AWS Glue Studio

AWS Glue Studio bietet eine visuelle Oberfläche zum Erstellen, Ausführen und Überwachen von Extract/Transform/Load (ETL-) Jobs in AWS Glue. Ein Job in AWS Glue besteht aus der Geschäftslogik, die Aufgaben zum Extrahieren, Transformieren und Laden (ETL) ausführt. Mit AWS Glue Studio können Sie Workflows zur Datentransformation visuell zusammenstellen und sie nahtlos auf AWS Glue der Apache Spark-basierten serverlosen ETL-Engine ausführen. Sie können Jobs erstellen, die Daten mithilfe einer drag-and-drop Schnittstelle zwischen verschiedenen Datenspeichern und Streams verschieben und transformieren, ohne Spark lernen oder Code schreiben zu müssen.

Ein AWS Glue-Auftrag kapselt ein Skript ein, das eine Verbindung zu den Quelldaten herstellt, verarbeitet es und schreibt es dann in Ihr Datenziel. In der Regel führt ein Auftrag Extraktions-, Transformations- und Ladeskripts (Extract, Transform and Load, ETL) aus. Aufträge können Skripte ausführen, die für Apache Spark und Ray-Laufzeitumgebungen entwickelt wurden. Jobs können auch allgemeine Python-Skripte (Python-Shell-Jobs) ausführen. AWS GlueTrigger können Jobs auf der Grundlage eines Zeitplans oder Ereignisses oder bei Bedarf starten. Sie können Auftragsausführungen überwachen, um mehr über Laufzeitmetriken wie Bearbeitungsstatus, Dauer und Startzeit zu erfahren.

Sie können von AWS Glue generierte Skripts verwenden oder eigene Skripts bereitstellen. Mit einem Quellschema und einer Zielposition oder einem Schema kann der AWS Glue Studio Codegenerator automatisch ein Apache Spark-API-Skript (PySpark) erstellen. Sie können dieses Skript als Ausgangspunkt verwenden und es bearbeiten, um Ihre Ziele zu erreichen.

AWS Glue kann Ausgabedateien in mehreren Datenformaten schreiben. Jeder Auftragstyp unterstützt möglicherweise unterschiedliche Ausgabeformate. Für einige Datenformate können gängige Komprimierungsformate geschrieben werden.

AWS Glue Jobs in der AWS Konsole verwalten

Um bestehende Jobs anzusehen, melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/. Wählen Sie anschließend die Registerkarte Jobs (Aufträge) in AWS Glue aus. Die Aufträge-Liste zeigt den Speicherort des Skripts, das mit jedem Auftrag verknüpft ist, den Zeitpunkt der letzten Auftragsänderung und die Textmarkenoption des aktuellen Auftrags an.

Sie können Aufträge im Abschnitt ETL der AWS Glue-Konsole erstellen. Beim Erstellen eines neuen Auftrags bzw. nachdem Sie Ihren Auftrag gespeichert haben, können Sie mit AWS Glue Studio Ihre ETL-Aufträge ändern. Dazu bearbeiten Sie die Knoten im visuellen Editor bzw. das Auftragsskript im Entwicklermodus. Sie können auch Knoten im visuellen Editor hinzufügen und entfernen und komplexere ETL-Aufträge erstellen.

Nächste Schritte zum Erstellen eines Auftrags in AWS Glue Studio

Sie verwenden den visuellen Auftragseditor, um Knoten für Ihren Auftrag zu konfigurieren. Jeder Knoten stellt eine Aktion dar, z. B. das Lesen von Daten vom Quellspeicherort oder das Anwenden einer Transformation auf die Daten. Jeder Knoten, den Sie Ihrem Auftrag hinzufügen, verfügt über Eigenschaften, die Informationen zum Datenspeicherort oder zur Transformation bereitstellen.

Die nächsten Schritte zum Erstellen und Verwalten Ihrer Jobs sind:

- Starten von Visual ETL-Jobs in AWS Glue Studio
- Anzeigen des Auftragsskripts
- Ändern der Auftragseigenschaften
- Speichern des Auftrags
- Starten einer Auftragsausführung
- Anzeigen von Informationen zu den letzten Auftragsausführungen
- Zugriff auf das Dashboard für die Auftragsüberwachung

Erstellen Sie visuelle ETL-Flows mit Amazon SageMaker

Mit einem Amazon SageMaker Unified Studio-Workflow können Sie eine Reihe von Aufgaben in Amazon SageMaker Unified Studio einrichten und ausführen. Amazon SageMaker Unified Studio-Workflows verwenden Apache Airflow, um Datenverarbeitungsverfahren zu modellieren und Ihre Amazon SageMaker Unified Studio-Codeartefakte zu orchestrieren. Weitere Informationen finden Sie unter Workflows in Amazon SageMaker Unified Studio verwenden.

Starten von Visual ETL-Jobs in AWS Glue Studio

Sie können die einfache visuelle Oberfläche verwenden in AWS Glue Studio um Ihre ETL-Jobs zu erstellen. Verwenden Sie die Seite Jobs (Aufträge), um neue Aufträge zu erstellen. Sie können auch

einen Skripteditor oder ein Notizbuch verwenden, um direkt mit Code in der AWS Glue Studio ETL-Jobskript.

Auf der Seite Jobs können Sie alle Jobs sehen, die Sie entweder mit AWS Glue Studio or AWS Glue. Auf dieser Seite können Sie Ihre Jobs anzeigen, verwalten und ausführen.

Sehen Sie sich auch das Blog-Tutorial an an einem anderen Beispiel, wie man ETL-Jobs erstellt mit AWS Glue Studio.

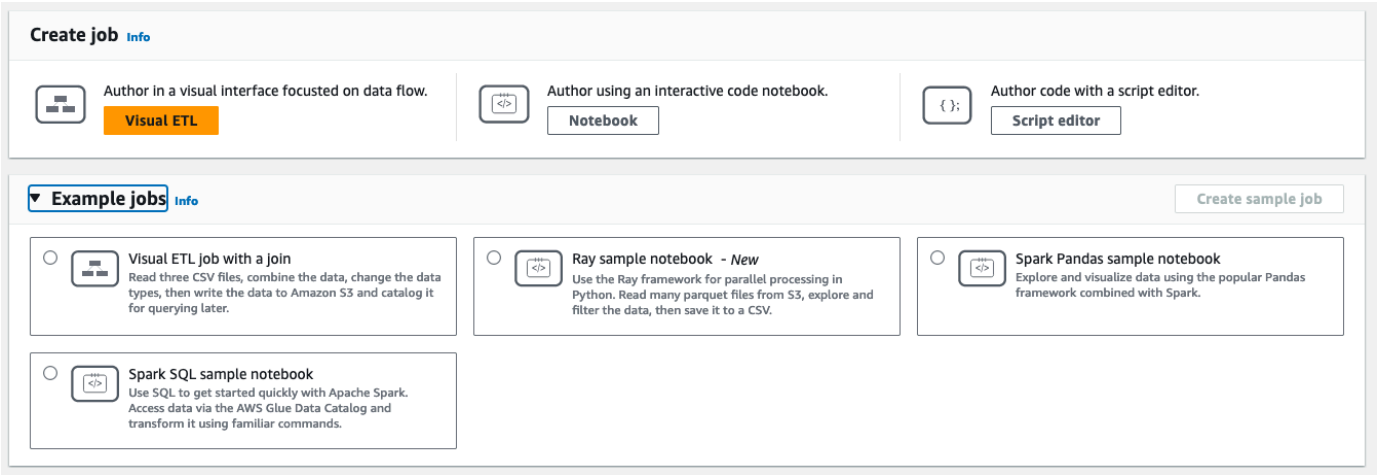
Jobs starten in AWS Glue Studio

AWS Glue ermöglicht es Ihnen, einen Job über eine visuelle Oberfläche, ein interaktives Code-Notizbuch oder mit einem Skripteditor zu erstellen. Sie können einen Auftrag starten, indem Sie auf eine der Optionen klicken, oder einen neuen Auftrag basierend auf einem Musterauftrag erstellen.

Beispielaufträge erstellen eine Auftrag mit dem Tool Ihrer Wahl. Mit Beispieljobs können Sie beispielsweise einen visuellen ETL-Job erstellen, der CSV-Dateien zu einer Katalogtabelle zusammenfügt, einen Job in einem interaktiven Code-Notizbuch mit AWS Glue für Ray oder AWS Glue für Spark, wenn Sie mit Pandas arbeiten, oder erstellen Sie einen Job in einem interaktiven Code-Notizbuch mit SparkSQL.

Einen Job erstellen in AWS Glue Studio von Grund auf

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Studio Konsole bei https://console.aws.amazon.com/gluestudio/.
2. Wählen Sie ETL-Aufträge aus dem Navigationsbereich.
3. Wählen Sie im Bereich Auftrag erstellen eine Konfigurationsoption für Ihren Auftrag aus.



Optionen zum Erstellen eines Auftrags von Grund auf:

- Visuelles ETL – verfassen in einer visuellen Benutzeroberfläche mit Fokus auf den Datenablauf
- Verfassen mithilfe eines interaktiven Code-Notebooks – Verfassen Sie Aufträge interaktiv in einer Benutzeroberfläche, die auf Jupyter Notebooks basiert

  Wenn Sie diese Option auswählen, müssen Sie zusätzliche Informationen bereitstellen, bevor Sie eine Sitzung zur Erstellung eines Notebooks erstellen. Weitere Informationen zum Angeben dieser Informationen finden Sie unter Erste Schritte mit Notebooks in AWS Glue Studio.
- Verfassen von Code mit einem Skript-Editor – Wenn Sie mit dem Programmieren und Schreiben von ETL-Skripts vertraut sind, wählen Sie diese Option, um einen neuen Spark ETL-Auftrag zu erstellen. Wählen Sie die Engine (Python-Shell, Ray, Spark (Python) oder Spark (Scala). Wählen Sie dann Neu starten oder Skript hochladen. Hochladen eines vorhandenen Skripts aus einer lokalen Datei. Wenn Sie den Skripteditor verwenden, können Sie den visuellen Auftragseditor zum Designen oder Bearbeiten Ihres Auftrags nicht verwenden.

  Ein Spark-Job wird in einer Apache Spark-Umgebung ausgeführt, die verwaltet wird von AWS Glue. Standardmäßig werden neue Skripte in Python codiert. Informationen zum Schreiben eines neuen Scala-Skripts finden Sie unter Erstellen und Bearbeiten von Scala-Skripten in AWS Glue Studio.

Einen Job erstellen in AWS Glue Studio aus einem Beispieljob

Sie können einen Auftrag anhand eines Beispielauftrags auswählen. Wählen Sie im Bereich Beispielaufträge einen Beispielauftrag und wählen Sie dann Beispielauftrag erstellen aus. Die Erstellung eines Beispielauftrags aus einer der Optionen stellt eine schnelle Vorlage bereit, mit der Sie arbeiten können.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Studio Konsole bei https://console.aws.amazon.com/gluestudio/.
2. Wählen Sie ETL-Aufträge aus dem Navigationsbereich.
3. Wählen Sie eine Option zum Erstellen eines Auftrags aus einem Beispielauftrag:

- Visueller ETL-Auftrag zum Zusammenführen mehrerer Quellen – Lesen Sie drei CSV-Dateien, kombinieren Sie die Daten, ändern Sie die Datentypen, schreiben Sie die Daten dann in Amazon S3 und katalogisieren Sie sie zur späteren Abfrage.
- Spark-Notebook mit Pandas – Erkunden und visualisieren Sie Daten mit dem beliebten Pandas-Framework in Kombination mit Spark.
- Spark-Notebook mit SQL – Verwenden Sie SQL, um schnell mit Apache Spark zu beginnen. Greifen Sie auf Daten über die zu AWS Glue Datenkatalog und transformieren Sie ihn mit vertrauten Befehlen.

4. Wählen Sie Beispielauftrag erstellen aus.

Features des Auftragseditors

Der Auftragseditor enthält die folgenden Features zum Erstellen und Bearbeiten von Aufträgen.

- Ein visuelles Diagramm des Auftrags mit einem Knoten für jede Auftragsaufgabe: Datenquellknoten zum Lesen der Daten; Transformationsknoten zum Ändern der Daten; Datenzielknoten zum Schreiben der Daten.

  Sie können die Eigenschaften der einzelnen Knoten im Auftragsdiagramm ansehen und konfigurieren. Sie können auch das Schema und die Beispieldaten für die Knoten im Auftragsdiagramm ansehen. Mit diesen Features können Sie feststellen, ob Ihr Auftrag die Daten korrekt ändert und transformiert, ohne dafür den Auftrag ausführen zu müssen.
- Eine Registerkarte zum Anzeigen und Bearbeiten von Skripts, auf der Sie den für Ihren Auftrag generierten Code ändern können;
- Eine Registerkarte mit den Jobdetails, auf der Sie eine Vielzahl von Einstellungen konfigurieren können, um die Umgebung anzupassen, in der Ihr AWS Glue Der ETL-Job wird ausgeführt.
- Die Registerkarte „Runs (Ausführungen)“ mit den aktuellen und vorherigen Auftragsausführungen, dem Status der Auftragsausführung und Protokollen für die Auftragsausführung;
- Die Registerkarte „Datenqualität“, auf der Sie Regeln zur Datenqualität für Ihren Auftrag anwenden können.
- Die Registerkarte „Schedules (Zeitpläne)“, auf der Sie die Startzeit für Ihren Auftrag konfigurieren oder wiederkehrende Auftragsausführungen einrichten können.
- Die Registerkarte „Versionsverwaltung“, auf der Sie einen Git-Dienst für Ihren Auftrag konfigurieren können.

Verwenden von Schemavorschauen im visuellen Auftragseditor

Beim Erstellen oder Bearbeiten eines Auftrags können Sie die auf der Registerkarte Output schema (Ausgabeschema) das Schema für Ihre Daten anzeigen.

Bevor Sie das Schema sehen können, benötigt der Auftragseditor Berechtigungen für den Zugriff auf die Datenquelle. Sie können eine IAM-Rolle auf der Registerkarte „Job details (Auftragsdetails)“ des Editors oder auf der Registerkarte (Output schema) Ausgabeschema eines Knotens angeben. Wenn die IAM-Rolle über alle erforderlichen Berechtigungen für den Zugriff auf die Datenquelle verfügt, können Sie das Schema auf der Registerkarte Output schema (Ausgabeschema) eines Knotens anzeigen.

Verwenden von Datenvorschauen im visuellen Auftragseditor

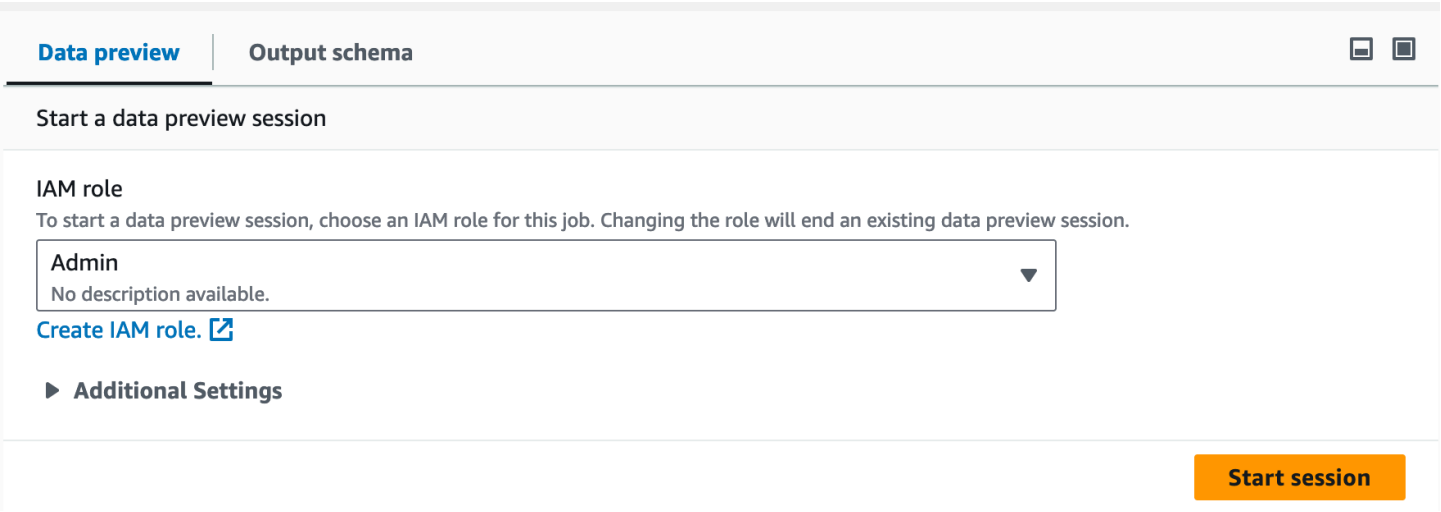
Mithilfe der Datenvorschau können Sie Ihren Auftrag mit einer Auswahl Ihrer Daten erstellen und testen, ohne den Auftrag immer wieder ausführen zu müssen. Die Datenvorschau bietet Ihnen folgende Möglichkeiten:

- Sie können eine IAM-Rolle testen, um sicherzustellen, dass Sie Zugriff auf die Datenquellen bzw. Datenziele haben.
- Sie können überprüfen, ob die Transformation die Daten wie beabsichtigt ändert. Wenn Sie beispielsweise eine Filter-Transformation verwenden, können Sie überprüfen, ob der Filter die richtige Teilmenge der Daten auswählt.
- Sie können Ihre Daten überprüfen. Wenn Ihr Datensatz Spalten mit Werten mehrerer Typen enthält, zeigt die Datenvorschau eine Liste von Tupeln für diese Spalten an. Jedes Tupel enthält den Datentyp und seinen Wert.

> **Note**
>
> Wenn Sie eine Datenvorschausitzung und einen benutzerdefinierten SQL- oder benutzerdefinierten Codeknoten verwenden, führt die Datenvorschausitzung den SQL- oder Codeblock unverändert für den gesamten Datensatz aus.

Beim Erstellen oder Bearbeiten eines Auftrags können Sie mit der Registerkarte Datenvorschau unterhalb der Auftragsleinwand ein Beispiel Ihrer Daten anzeigen. Eine neue Datenvorschau-Sitzung

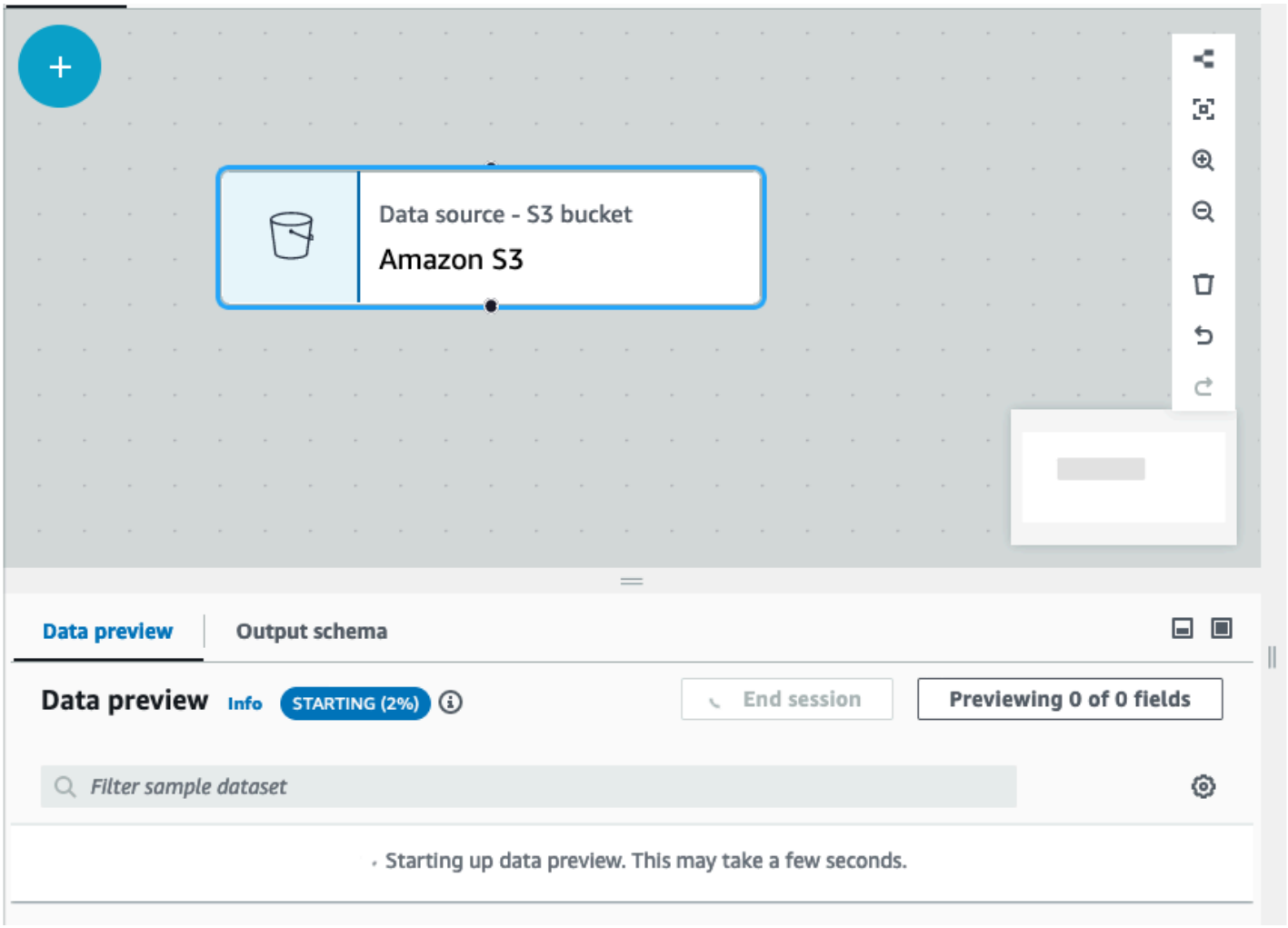
wird automatisch gestartet, wenn die Rolle für den Auftrag bereits konfiguriert ist oder eine Standard-IAM-Rolle im Konto eingerichtet wurde. Wenn noch keine Rolle konfiguriert wurde, können Sie eine Sitzung starten, indem Sie die Rolle auswählen.



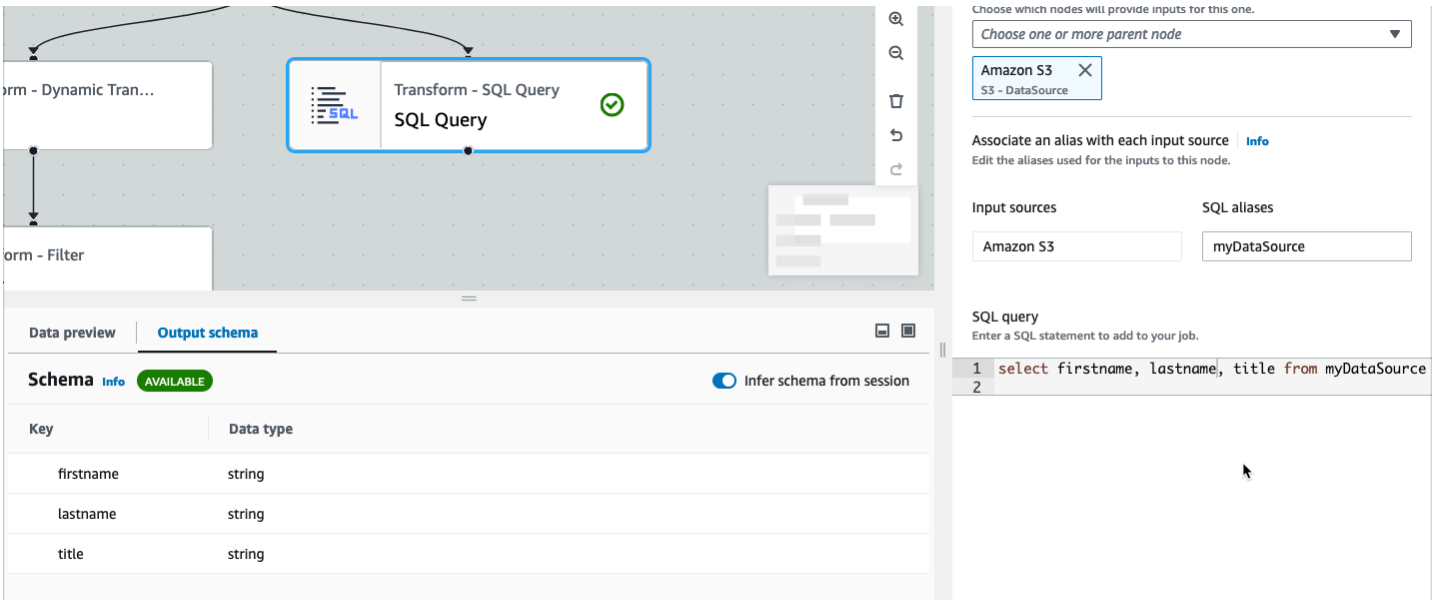
> **Note**
>
> Die Rolle, die Sie für die Datenvorschau-Sitzung auswählen, wird auch für den Auftrag verwendet.

Sie können den Status und den Fortschritt Ihrer Sitzung sowie die Sitzungsdetails aufrufen, indem Sie auf das Infosymbol klicken.

Wenn die Sitzung bereit ist, AWS Glue Studio lädt die Daten für den ausgewählten Knoten. Sie können den Status % abgeschlossen beobachten.



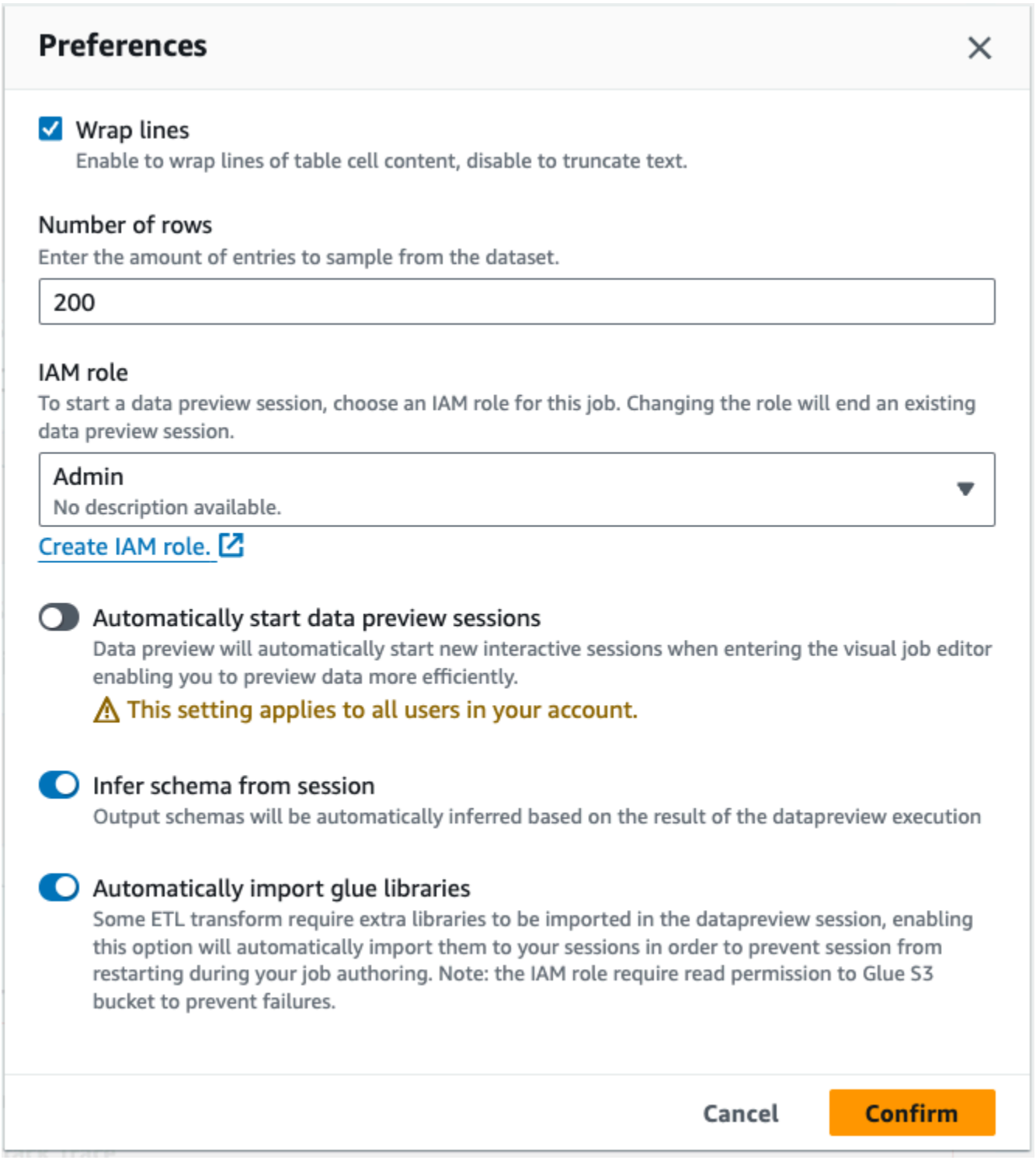
Während Sie Ihren visuellen Job verfassen, AWS Glue Studio aktualisiert automatisch das Schema für den ausgewählten Knoten, wenn Sie auf der Registerkarte Ausgabeschema die Option Schema aus Sitzung ableiten aktivieren.



Konfigurieren Sie die Einstellungen für die Datenvorschau wie folgt:

Wählen Sie das Einstellungssymbol (ein Zahnrad), um Voreinstellungen für die Datenvorschau zu treffen. Diese Einstellungen gelten für alle Knoten im Auftragsdiagramm. Sie haben folgende Möglichkeiten:

- Wählen Sie, dass der Text von einer Zeile in die nächste umgebrochen wird. Diese Option ist standardmäßig aktiviert.
- Ändern Sie die Anzahl der Zeilen (Standard: 200).
- Wählen Sie eine IAM-Rolle aus oder erstellen Sie bei Bedarf eine neue.
- Wählen Sie aus, dass automatisch eine neue Sitzung gestartet wird, wenn Sie einen Auftrag erstellen. Dadurch wird beim Erstellen von Aufträgen eine neue interaktive Sitzung gestartet. Diese Einstellung wird auf Kontoebene angewendet. Einmal eingestellt, gilt sie für alle Benutzer in Ihrem Konto, wenn Sie einen Auftrag bearbeiten.
- Wählen Sie aus, dass das Schema automatisch abgeleitet wird. Ausgabeschemas werden für den ausgewählten Knoten automatisch abgeleitet.
- Wählen Sie „Automatisch importieren“ AWS Glue Bibliotheken. Dies ist nützlich, da es verhindert, dass die Datenvorschau Sitzungen neu startet, wenn neue Transformationen hinzugefügt werden, die einen Sitzungsneustart erfordern.

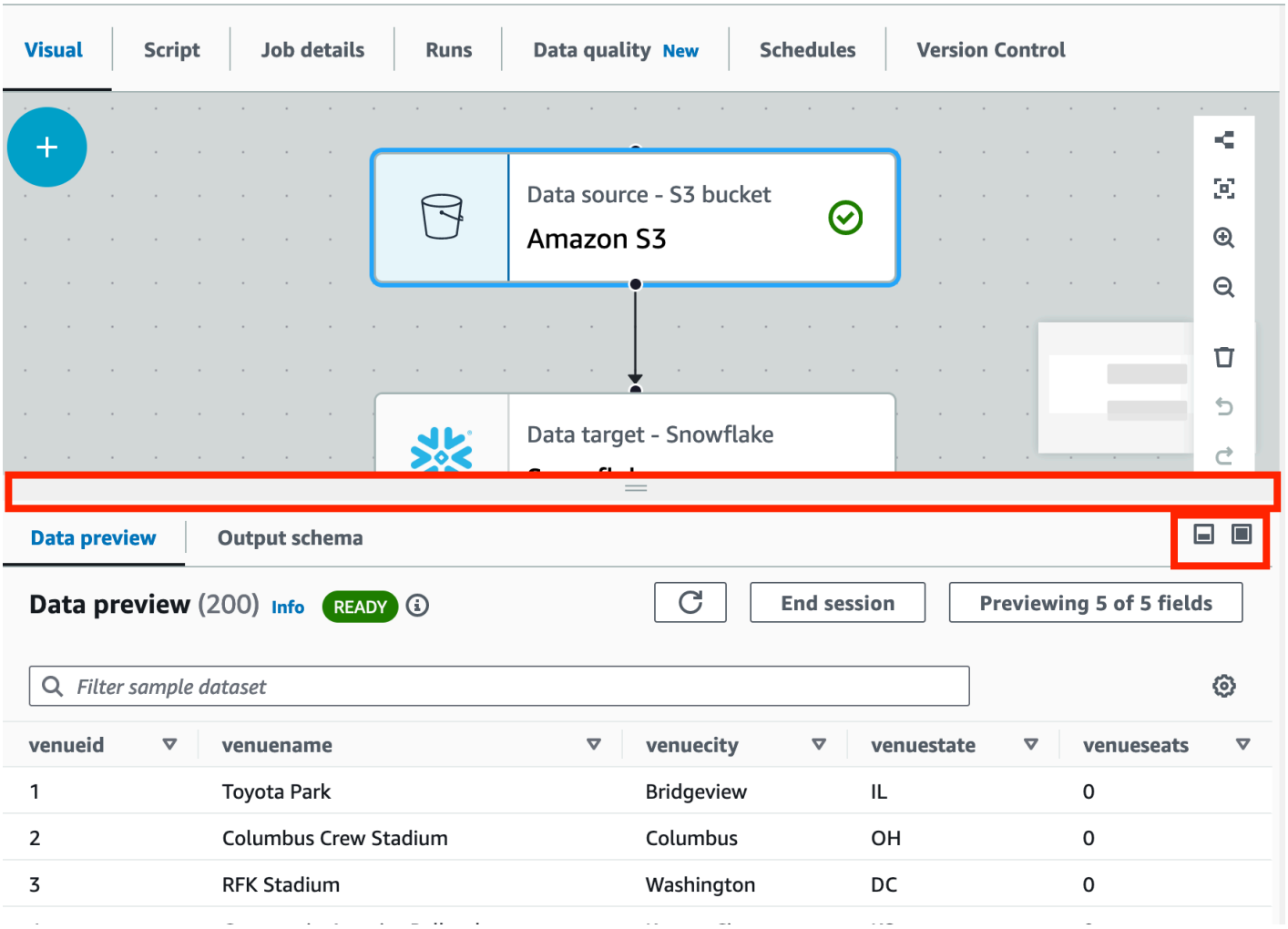


Weitere Features bieten Ihnen unter anderem folgende Möglichkeiten:

- Mit der Schaltfläche Previewing x of y fields (Vorschau von x von y Feldern) können Sie auswählen, welche Spalten (Felder) in der Vorschau angezeigt werden sollen. Bei der Standardeinstellung

zeigt der Auftragseditor in der Auftragsvorschau die ersten 5 Spalten Ihres Datensatzes an. Das lässt sich so ändern, dass entweder alle oder keine angezeigt werden (nicht empfohlen).

- Sie können sowohl horizontal als auch vertikal durch das Datenvorschaufenster scrollen.
- Mit der Schaltfläche „Maximieren“ können Sie die Registerkarte „Datenvorschau“ so ausdehnen, dass sie über dem Auftragsdiagramm liegt und die Daten und Datenstrukturen besser zu sehen sind. Verwenden Sie auf ähnliche Weise die Schaltfläche „Minimieren“, um die Registerkarte „Datenvorschau“ zu minimieren. Sie können auch den Rand des Fensters nach oben ziehen, um die Registerkarte Datenvorschau zu erweitern.



- Verwenden Sie Sitzung beenden, um die Datenvorschau zu beenden. Wenn Sie die Sitzung beenden, können Sie eine neue IAM-Rolle auswählen und zusätzliche Einstellungen festlegen (z. B. Einstellungen zum automatischen Starten einer neuen Sitzung ein- oder ausschalten, ein Schema ableiten oder importieren) AWS Glue Bibliotheken und starten Sie die Sitzung erneut.

Einschränkungen bei der Verwendung von Datenvorschauen

Bei Datenvorschauen bemerken Sie mitunter die folgenden Einschränkungen.

- Wenn Sie das erste Mal die Registerkarte „Data preview (Datenvorschau)“ aufrufen, müssen Sie eine IAM-Rolle auswählen. Diese Rolle muss über die erforderlichen Berechtigungen verfügen, um auf die Daten und andere Ressourcen zum Erstellen der Datenvorschau zuzugreifen.
- Nachdem Sie eine IAM-Rolle bereitgestellt haben, dauert es eine Weile, bis die Daten angezeigt werden können. Bei Datensätzen mit weniger als 1 GB kann dies bis zu einer Minute dauern. Bei einem großen Datensatz sollten Sie Partitionen verwenden, um die Ladezeit zu verkürzen. Am schnellsten geht das Laden von Daten direkt von Amazon S3.
- Bei einem sehr großen Datensatz kommt es zu einer Zeitüberschreitung, wenn das Abfragen der Daten für die Datenvorschau mehr als 15 Minuten dauert. Für Datenvorschauen ist ein IDLE-Timeout von 30 Minuten festgelegt. Um dieses Problem zu entschärfen, müssen Sie die Größe von Datensätzen für die Datenvorschau verkleinern.
- Standardmäßig werden die ersten 50 Spalten auf der Registerkarte „Datenvorschau“ angezeigt. Wenn die Spalten keine Datenwerte haben, erscheint eine Meldung, dass es keine Daten zum Anzeigen gibt. Sie können die Anzahl der für das Beispiel herangezogenen Zeilen erhöhen oder verschiedene Spalten auswählen, um Datenwerte zu sehen.
- Datenvorschauen sind aktuell nicht möglich bei Streaming-Datenquellen oder für Datenquellen mit benutzerdefinierten Konnektoren.
- Fehler bei einem Knoten wirken sich auf den gesamten Auftrag aus. Wenn ein Knoten einen Fehler bei der Datenvorschau hat, wird der Fehler auf allen Knoten angezeigt, bis Sie ihn korrigieren.
- Wenn Sie eine Datenquelle für den Auftrag ändern, müssen die untergeordneten Knoten dieser Datenquelle möglicherweise für das neue Schema aktualisiert werden. Wenn Sie beispielsweise über einen ApplyMapping Knoten verfügen, der eine Spalte ändert, und die Spalte in der Ersatzdatenquelle nicht vorhanden ist, müssen Sie den ApplyMapping Transformationsknoten aktualisieren.
- Wenn die SQL-Abfrage einen falschen Feldnamen verwendet, wird bei einem SQL-Abfrage-Transformationsknoten auf der Registerkarte „Data preview (Datenvorschau)“ ein Fehler angezeigt.

Generierung des Skript-Code

Wenn Sie den Visual Editor verwenden, um einen Job zu erstellen, wird der ETL-Code automatisch für Sie generiert. AWS Glue Studio erstellt ein funktionsfähiges und vollständiges Jobskript und speichert es an einem Amazon S3 S3-Speicherort.

Es gibt zwei Arten von Code, die generiert werden von AWS Glue Studio: die ursprüngliche oder klassische Version und eine neuere, optimierte Version. Standardmäßig wird der neue Codegenerator zum Erstellen des Jobskripts verwendet. Sie können ein Auftragsskript mit dem klassischen Codegenerator auf der Registerkarte Script generieren, indem Sie die Schaltfläche Generate classic script (Generiere klassisches Skript) zum Umschalten wählen.

Einige der Unterschiede in der neuen Version des generierten Codes sind:

- Große Kommentarblöcke werden dem Skript nicht mehr hinzugefügt
- Ausgabestrukturen im Code verwenden den Knotennamen, den Sie im visuellen Editor angeben. Im Klassenskript werden die Ausgabestrukturen einfach benannt `DataSource0`,`DataSource1`,`Transform0`,`Transform1`,`DataSink0`,`DataSink1`, usw.
- Lange Befehle sind auf mehrere Zeilen aufgeteilt, um nicht über die Seite scrollen zu müssen, um den gesamten Befehl anzuzeigen.

Neue Funktionen in AWS Glue Studio erfordert die neue Version der Codegenerierung und funktioniert nicht mit dem klassischen Codeskript. Sie werden aufgefordert, diese Aufträge zu aktualisieren, wenn Sie versuchen, sie auszuführen.

Transformieren Sie Daten mit AWS Glue verwaltete Transformationen

AWS Glue Studio bietet zwei Arten von Transformationen:

- AWS Glue-native Transformationen — stehen allen Benutzern zur Verfügung und werden verwaltet von AWS Glue.
- Benutzerdefinierte visuelle Transformationen — ermöglicht es Ihnen, Ihre eigenen Transformationen hochzuladen, um sie in zu verwenden AWS Glue Studio

AWS Glue verwaltete Datentransformationsknoten

AWS Glue Studio bietet eine Reihe integrierter Transformationen, mit denen Sie Ihre Daten verarbeiten können. Ihre Daten werden in einer Datenstruktur namens `DynamicFrame` von einem Knoten im Auftragsdiagramm zum anderen übergeben (eine Erweiterung für ein Apache Spark SQL `DataFrame`).

Im vorab ausgefüllten Diagramm für einen Job befindet sich zwischen den Datenquellen- und Datenzielknoten der Transformationsknoten Schema ändern. Sie können diesen

Transformationsknoten so konfigurieren, dass er die Daten ändert, oder zusätzliche Transformationen verwenden.

Die folgenden integrierten Transformationen sind verfügbar mit AWS Glue Studio:

- ChangeSchema: Ordnet Dateneigenschaftsschlüssel in der Datenquelle Dateneigenschaftsschlüsseln im Datenziel zu. Sie können Schlüssel umbenennen, die Datentypen für Schlüssel ändern und die Schlüssel auswählen, die aus dem Datensatz gelöscht werden sollen.
- SelectFields: Wählen Sie die Dateneigenschaftsschlüssel aus, die Sie behalten möchten.
- DropFields: Wählen Sie die Dateneigenschaftsschlüssel aus, die Sie löschen möchten.
- RenameField: Benennt einen einzelnen Dateneigenschaftsschlüssel um.
- Spigot: Schreiben Sie Beispiele der Daten in einen Amazon-S3-Bucket.
- Join: Führen Sie zwei Datensätze mit einer Vergleichsphrase für die angegebenen Dateneigenschaftsschlüssel zu einem Datensatz zusammen. Sie können innere, äußere, linke, rechte, linke Hälfte und linke Anti-Joins verwenden.
- Vereinigung: Kombinieren Sie Zeilen aus mehr als einer Datenquelle, die dasselbe Schema haben.
- SplitFields: Teilt Dateneigenschaftsschlüssel in zwei Teile auf`DynamicFrames`. Die Ausgabe ist eine Sammlung von `DynamicFrames`: Einer mit ausgewählten Dateneigenschaftsschlüsseln und einer mit den übrigen Dateneigenschaftsschlüsseln.
- SelectFromCollection: Wählen Sie einen `DynamicFrame` aus einer Sammlung von`DynamicFrames`. Die Ausgabe ist der ausgewählte `DynamicFrame`.
- FillMissingValues: Suchen Sie nach Datensätzen im Datensatz, bei denen Werte fehlen, und fügen Sie ein neues Feld mit einem vorgeschlagenen Wert hinzu, der durch Imputation bestimmt wird
- Filter: Teilen Sie ein Datensatz anhand einer Filterbedingung in zwei Datensätze auf.
- Leere Felder löschen: Entfernt Spalten aus dem Datensatz, wenn alle Werte in der Spalte „Null“ sind.
- Duplikate löschen: Entfernt Zeilen aus Ihrer Datenquelle, indem entweder ganze Zeilen abgeglichen oder Schlüssel angegeben werden.
- SQL: Geben Sie SparkSQL-Programmiercode in ein Texteingabefeld ein, um eine SQL-Abfrage zum Transformieren der Daten zu verwenden. Die Ausgabe ist ein einzelner `DynamicFrame`.
- Aggregierung: führt eine Berechnung (wie Durchschnitt, Summe, Min, Max) für ausgewählte Felder und Zeilen durch und erstellt ein neues Feld mit den neu berechneten Werten/dem neu berechneten Wert.
- Abflachen: Extrahiert Felder innerhalb von Strukturen in Felder der obersten Ebene.

- UUID: Fügt für jede Zeile eine Spalte mit einem universell eindeutigen Bezeichner hinzu.
- Identifikator: Fügt für jede Zeile eine Spalte mit einer numerischen ID hinzu.
- Zum Zeitstempel: Konvertiert eine Spalte in den Zeitstempeltyp.
- Zeitstempel formatieren: Konvertiert eine Zeitstempelspalte in eine formatierte Zeichenfolge.
- Bedingte Router-Transformation: Wenden Sie mehrere Bedingungen auf eingehende Daten an. Jede Zeile der eingehenden Daten wird anhand einer Gruppenfilterbedingung ausgewertet und zu der entsprechenden Gruppe verarbeitet.
- Transformation einer Verkettung von Spalten: Erstellen Sie eine neue Zeichenfolgenspalte unter Verwendung der Werte anderer Spalten mit einem optionalen Abstandszeichen.
- Transformation einer geteilten Zeichenfolge: Teilen Sie eine Zeichenfolge mithilfe eines regulären Ausdrucks in ein Array von Token auf, um zu definieren, wie die Aufteilung durchgeführt wird.
- Transformation von Array zu Spalten: Extrahieren Sie einige oder alle Elemente einer Spalte vom Typ Array in neue Spalten.
- Transformation „Aktuellen Zeitstempel hinzufügen“: Markieren Sie die Zeilen mit der Uhrzeit, zu der die Daten verarbeitet wurden. Dies ist für Prüfzwecke oder zum Verfolgen der Latenz in der Datenpipeline nützlich.
- Transformation „Zeilen zu Spalten pivotieren“: Aggregieren Sie eine numerische Spalte, indem Sie eindeutige Werte in ausgewählten Spalten rotieren, die zu neuen Spalten werden. Bei Auswahl mehrerer Spalten werden die Werte verkettet, um die neuen Spalten zu benennen.
- Transformation „Spalten zu Zeilen entpivotieren“: Konvertieren Sie Spalten in Werte neuer Spalten und erzeugen Sie eine Zeile für jeden eindeutigen Wert.
- Transformation zur automatischen Balance-Verarbeitung: Verteilen Sie die Daten besser unter den Mitarbeitern. Dies ist nützlich, wenn die Daten unausgeglichen sind oder aufgrund ihrer Quelle keine ausreichende Parallelverarbeitung möglich ist.
- Transformation für abgeleitete Spalten: Definieren Sie eine neue Spalte auf der Grundlage einer mathematischen Formel oder eines SQL-Ausdrucks, in der Sie andere Spalten in den Daten sowie Konstanten und Literale verwenden können.
- Nachschlage-Transformation: Fügen Sie Spalten aus einer definierten Katalogtabelle hinzu, wenn die Schlüssel mit den definierten Nachschlagespalten in den Daten übereinstimmen.
- Transformation „Matrix auflösen“ oder „In Zeilen zuordnen“: Extrahieren Sie Werte aus einer verschachtelten Struktur in einzelne Zeilen, die einfacher zu bearbeiten sind.
- Transformation für den Datensatzabgleich: Rufen Sie eine vorhandene Transformation zur Datenklassifizierung durch Machine Learning zum Datensatzabgleich auf.

- Transformation zum Entfernen von Nullzeilen: Entfernen Sie Zeilen aus dem Datensatz, deren Spalten alle null oder leer sind.
- Transformation zum Analysieren von JSON-Spalten: Analysieren Sie eine Zeichenfolgenspalte mit JSON-Daten und konvertieren Sie sie in eine Struktur- oder Array-Spalte, je nachdem, ob es sich bei JSON um ein Objekt oder ein Array handelt.
- Transformation zum Extrahieren des JSON-Pfads: Extrahieren Sie neue Spalten aus einer JSON-Zeichenfolgenspalte.
- Zeichenkettenfragmente aus einem regulären Ausdruck extrahieren: Extrahieren Sie Zeichenfolgenfragmente mithilfe eines regulären Ausdrucks und erstellen Sie daraus eine neue Spalte oder mehrere Spalten, wenn Sie Regex-Gruppen verwenden.
- Custom transform (benutzerdefinierte Transformation): Geben Sie Programmiercode in ein Texteingabefeld ein, um benutzerdefinierte Transformationen zu verwenden. Die Ausgabe ist eine Sammlung von `DynamicFrames`.

Verwendung eines Datenvorbereitungsrezepts in AWS Glue Studio

Mit der Transformation für Datenvorbereitungsrezepte können Sie mithilfe einer interaktiven Erstellungsoberfläche im Rasterstil ein völlig neues Datenvorbereitungsrezept erstellen. Sie können damit auch ein vorhandenes AWS Glue DataBrew Rezept importieren und es anschließend bearbeiten. AWS Glue Studio

Der Knoten Datenvorbereitungsrezept ist im Bereich Ressource verfügbar. Sie können den Knoten Datenvorbereitungsrezept mit einem anderen Knoten im visuellen Workflow verbinden, unabhängig davon, ob es sich um einen Datenquellenknoten oder einen anderen Transformationsknoten handelt. Nachdem Sie ein AWS Glue DataBrew Rezept und eine Version ausgewählt haben, sind die angewendeten Schritte im Rezept auf der Registerkarte mit den Knoteneigenschaften sichtbar.

Voraussetzungen

- Wenn Sie ein AWS Glue DataBrew Rezept importieren, verfügen Sie über die erforderlichen IAM-Berechtigungen, wie unter beschrieben Importiere ein AWS Glue DataBrew Rezept in AWS Glue Studio.
- Eine Datenvorschau-Sitzung muss erstellt werden.

Einschränkungen

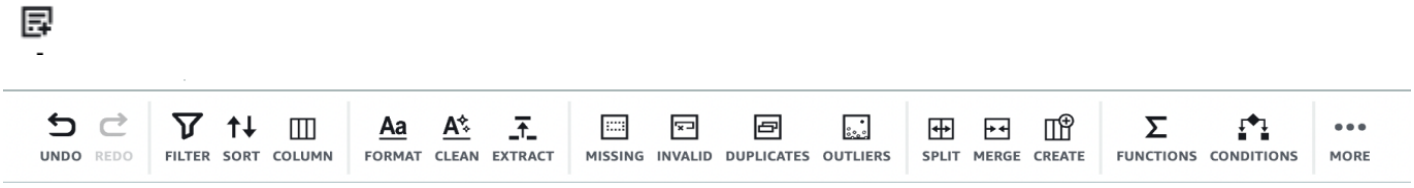
- AWS Glue DataBrew Rezepte werden nur in kommerziellen DataBrew Regionen unterstützt.

- Nicht alle AWS Glue DataBrew Rezepte werden unterstützt von AWS GlueEinige Rezepte können nicht in ausgeführt werden. AWS Glue Studio.
  - Rezepte mit `UNION` und `JOIN` Transformationen werden jedoch nicht unterstützt AWS Glue Studio verfügt bereits über die Transformationsknoten „Join“ und „Union“, die vor oder nach einem Data Preparation Recipe Node verwendet werden können.
- Data Preparation Recipe-Knoten werden für Jobs unterstützt, die mit beginnen AWS Glue Version 4.0. Diese Version wird automatisch ausgewählt, nachdem dem Auftrag ein Knoten für ein Datenvorbereitungsrezept hinzugefügt wurde.
- Knoten für ein Datenvorbereitungsrezept benötigen Python. Dies wird automatisch festgelegt, wenn der Knoten für ein Datenvorbereitungsrezept zum Auftrag hinzugefügt wird.
- Wenn Sie dem Visual Graph einen neuen Data Preparation Recipe Node hinzufügen, wird Ihre Datenvorschau-Sitzung automatisch mit den richtigen Bibliotheken neu gestartet, um den Data Preparation Recipe Node zu verwenden.
- Die folgenden Transformationen werden für den Import oder die Bearbeitung in einem Data Preparation Recepe-Node nicht unterstützt: `GROUP_BY`,`PIVOT`,`UNPIVOT`, und`TRANSPOSE`.

Weitere Features

Wenn Sie die Transformation „Datenvorbereitungsrezept“ ausgewählt haben, können Sie nach der Auswahl von „Rezept erstellen“ weitere Aktionen ausführen.

- Schritt hinzufügen — Sie können einem Rezept nach Bedarf weitere Schritte hinzufügen, indem Sie auf das Symbol „Schritt hinzufügen“ klicken oder die Werkzeugleiste im Vorschaufenster verwenden, indem Sie eine Aktion auswählen.



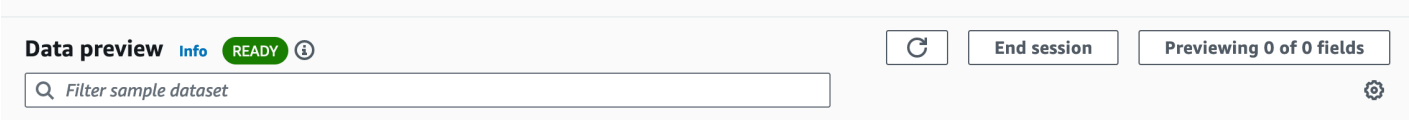
- Rezept importieren — wählen Sie Mehr und dann Rezept importieren, um es in Ihrem AWS Glue Studio Job zu verwenden.

- Als YAML herunterladen — Wählen Sie Mehr und dann Als YAML herunterladen, um Ihr Rezept herunterzuladen und außerhalb von zu speichern. AWS Glue Studio
- Als JSON herunterladen — wähle Mehr und dann Als JSON herunterladen, um dein Rezept herunterzuladen und außerhalb von zu speichern. AWS Glue Studio
- Rezeptschritte rückgängig machen und wiederholen — Sie können Rezeptschritte im Vorschaufenster rückgängig machen und wiederholen, wenn Sie mit Daten im Raster arbeiten.

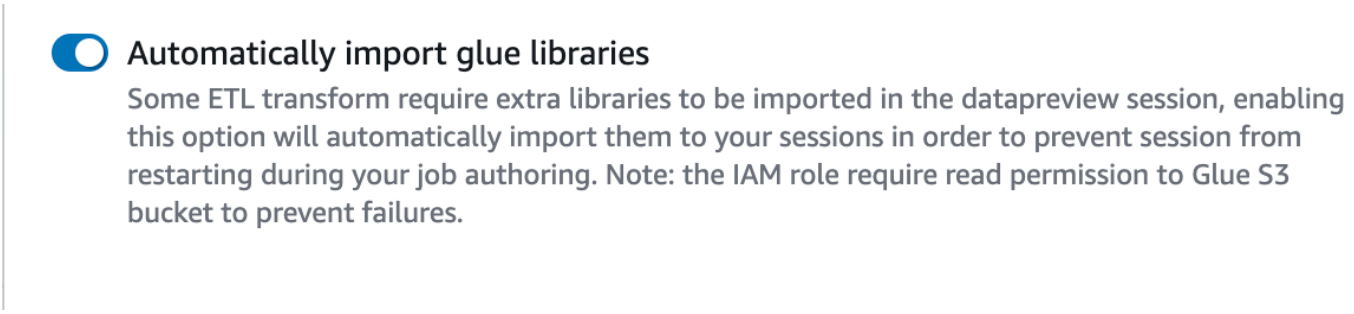
Erstellen und Ausführen von Rezepten für die Datenvorbereitung in einem visuellen AWS Glue ETL-Job

In diesem Szenario können Sie Rezepte für die Datenvorbereitung erstellen, ohne sie zuerst in erstellen zu müssen DataBrew. Bevor Sie mit dem Verfassen von Rezepten beginnen können, müssen Sie:

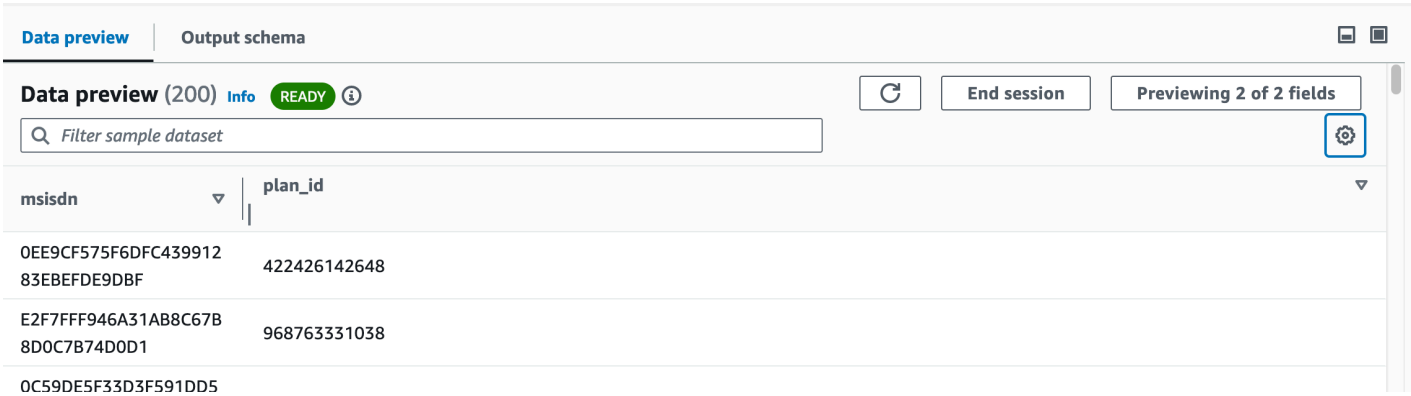
- Eine aktive Datenvorschau-Sitzung muss ausgeführt werden. Wenn die Datenvorschau-Sitzung BEREIT ist, wird Author Recipe aktiv und Sie können mit der Erstellung oder Bearbeitung Ihres Rezepts beginnen.



- Stellen Sie sicher, dass der Schalter für Glue-Bibliotheken automatisch importieren aktiviert ist.

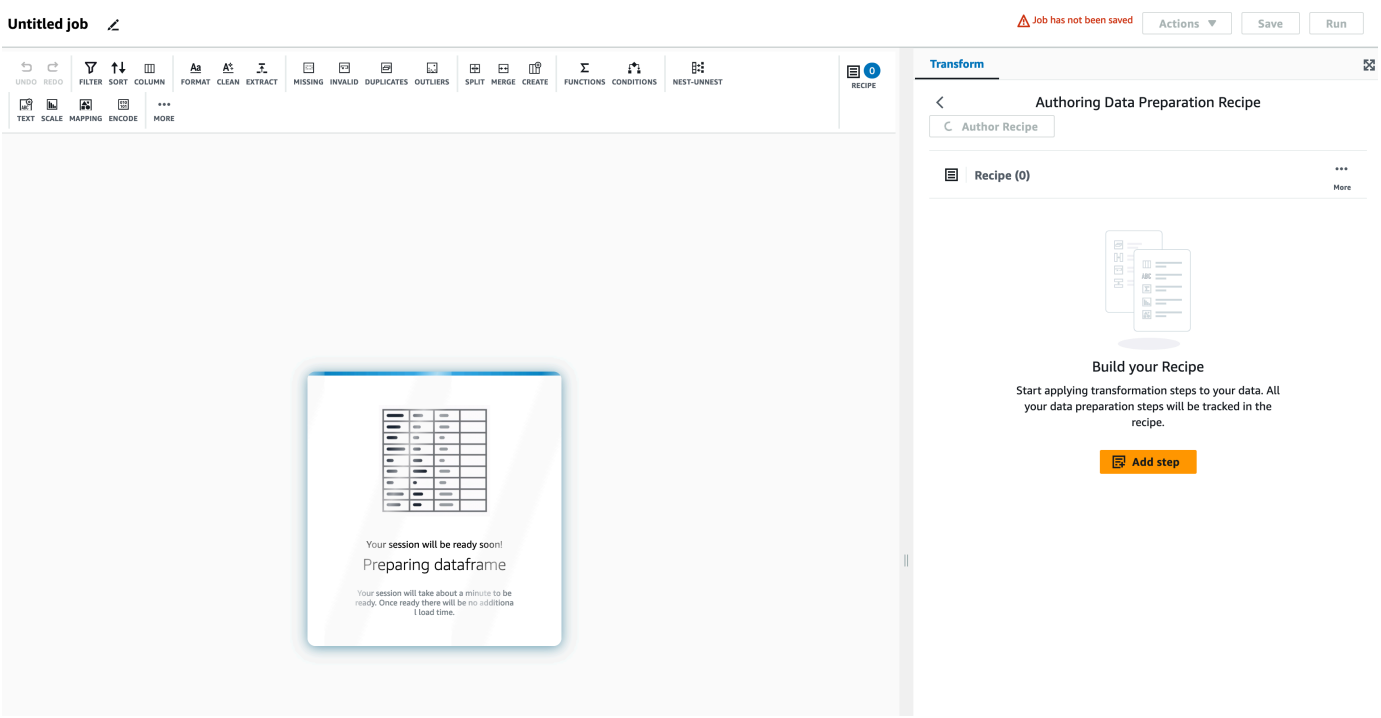


Sie können dies tun, indem Sie im Bereich Datenvorschau auf das Zahnradsymbol klicken.



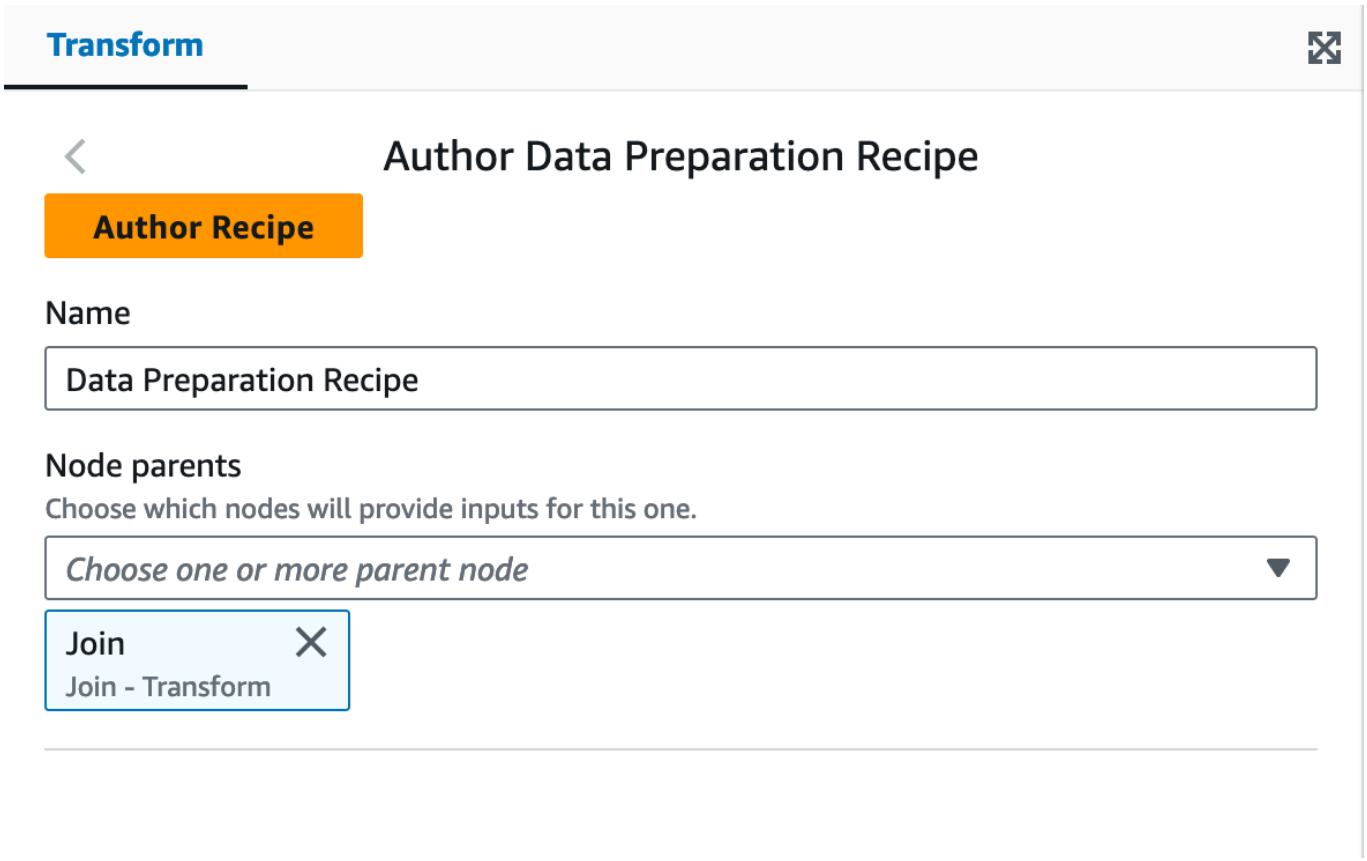
So erstellen Sie ein Rezept für die Datenvorbereitung in AWS Glue Studio:

1. Fügen Sie die Transformation für das Datenvorbereitungsrezept zu Ihrer Job-Canvas hinzu. Ihre Transformation sollte mit einem übergeordneten Datenquellenknoten verbunden sein. Wenn Sie den Knoten Data Preparation Recipe hinzufügen, wird der Knoten mit den richtigen Bibliotheken neu gestartet, und Sie werden sehen, dass der Datenrahmen vorbereitet wird.

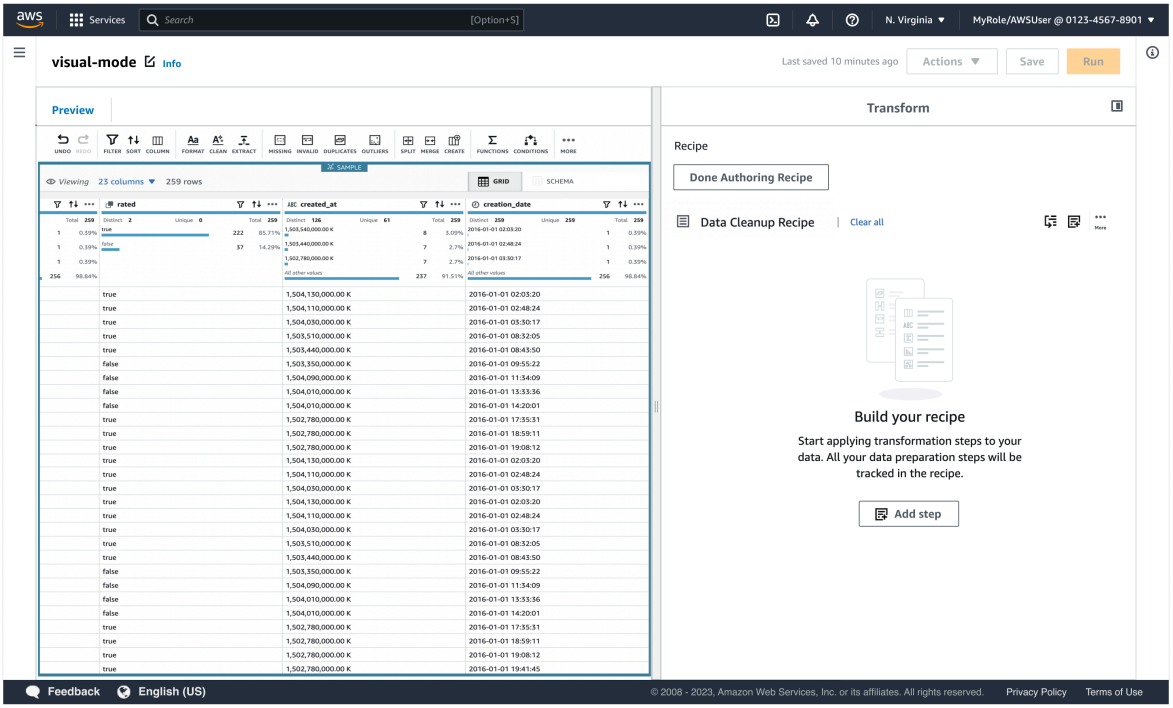


2. Sobald die Datenvorschau-Sitzung fertig ist, werden die Daten mit allen zuvor angewendeten Schritten am unteren Bildschirmrand angezeigt.

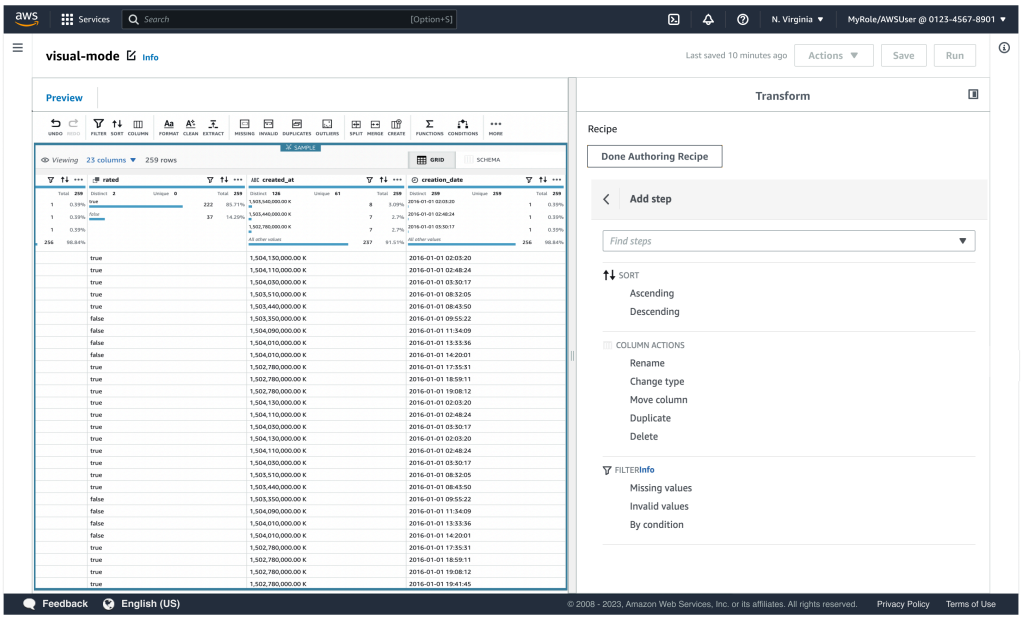
3. Wählen Sie Autorenrezept. Auf diese Weise können Sie ein neues Rezept in beginnen AWS Glue Studio.



4. Geben Sie im Bedienfeld „Transformieren“ rechts neben der Job-Canvas einen Namen für Ihr Datenvorbereitungsrezept ein.
5. Auf der linken Seite wird die Arbeitsfläche durch eine Rasteransicht Ihrer Daten ersetzt. Auf der rechten Seite ändert sich das Transformationsfenster und zeigt dir deine Rezeptschritte. Wählen Sie Schritt hinzufügen, um den ersten Schritt zu Ihrem Rezept hinzuzufügen.

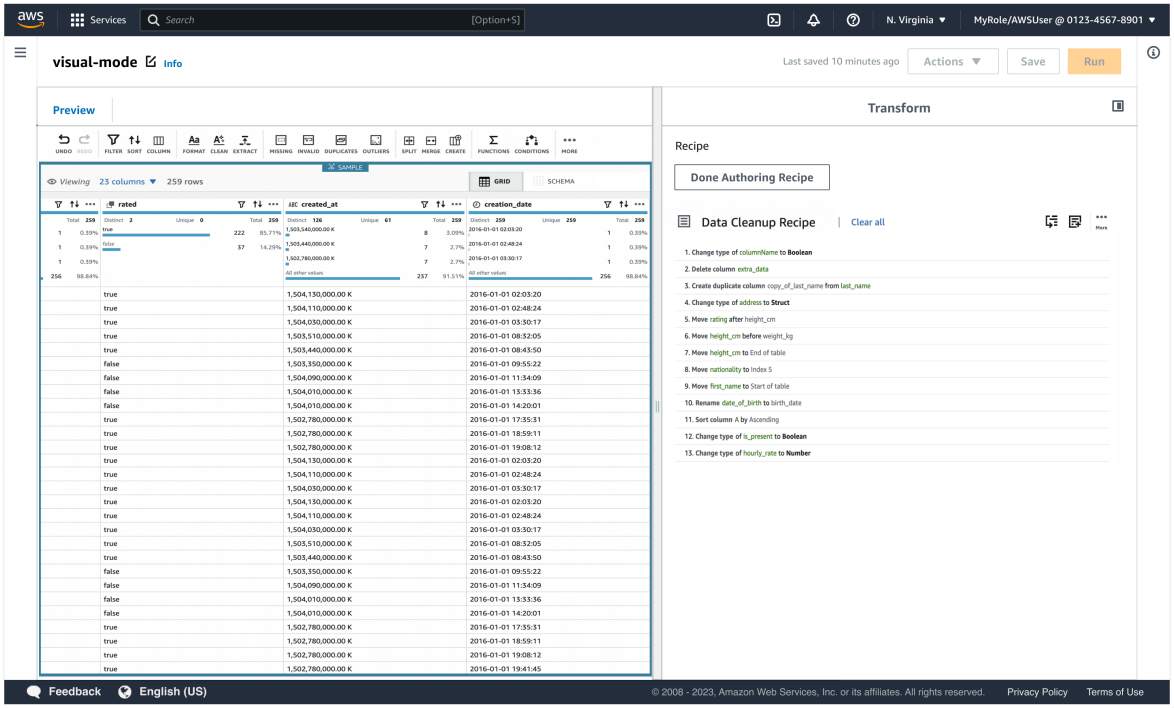


6. Wählen Sie im Bedienfeld „Transformieren“ aus, ob Sie die Spalte sortieren, eine Aktion für die Spalte ausführen und Werte filtern möchten. Wählen Sie beispielsweise „Spalte umbenennen“.



7. Im Transformationsfenster auf der rechten Seite können Sie mit den Optionen zum Umbenennen einer Spalte die Quellspalte auswählen, die Sie umbenennen möchten, und den neuen Spaltennamen eingeben. Sobald Sie dies getan haben, wählen Sie „Anwenden“.

   Sie können für jeden Schritt eine Vorschau anzeigen, einen Schritt rückgängig machen, die Schritte neu anordnen und eines der Aktionssymbole wie Filtern, Sortieren, Teilen, Zusammenführen usw. verwenden. Wenn Sie Aktionen im Datenraster ausführen, werden die Schritte dem Rezept im Bedienfeld „Transformieren“ hinzugefügt.

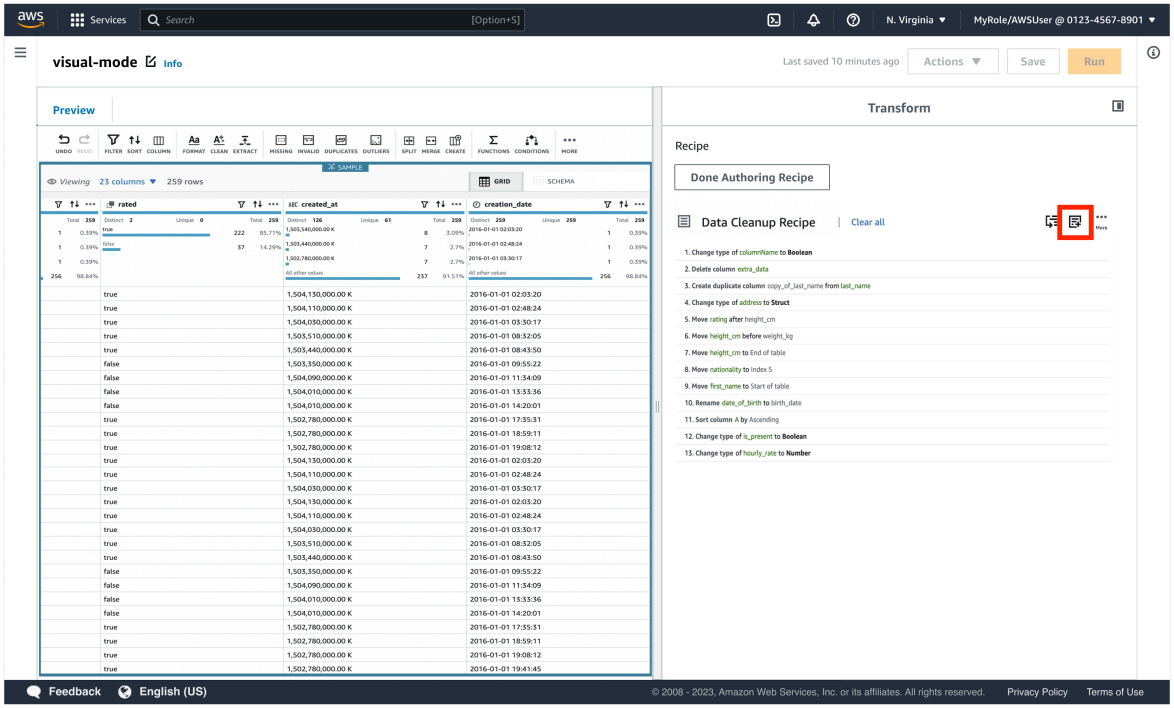


Wenn Sie eine Änderung vornehmen müssen, können Sie dies im Vorschaufenster tun, indem Sie eine Vorschau des Ergebnisses jedes Schritts anzeigen, einen Schritt rückgängig machen und die Schritte neu anordnen. Zum Beispiel:

- Schritt rückgängig machen/wiederholen — Machen Sie einen Schritt rückgängig, indem Sie auf das Rückgängig-Symbol klicken. Sie können einen Schritt wiederholen, indem Sie auf das Symbol „Wiederherstellen“ klicken.

- Schritt neu anordnen — Wenn Sie einen Schritt neu anordnen, AWS Glue Studio wird jeder Schritt validiert und Sie werden darüber informiert, ob der Schritt ungültig ist.

8. Sobald du einen Schritt angewendet hast, zeigt dir das Bedienfeld „Transformieren“ alle Schritte in deinem Rezept an. Sie können alle Schritte löschen, um von vorne zu beginnen, weitere Schritte hinzufügen, indem Sie auf das Symbol „Hinzufügen“ klicken, oder „Fertiges Authoring-Rezept“ wählen.



9. Wähle oben rechts auf dem Bildschirm Speichern aus. Ihre Rezeptschritte werden erst gespeichert, wenn Sie Ihren Job gespeichert haben.

Importiere ein AWS Glue DataBrew Rezept in AWS Glue Studio

In AWS Glue DataBrew besteht ein Rezept aus einer Reihe von Schritten zur Datentransformation. AWS Glue DataBrew Rezepte schreiben vor, wie Daten transformiert werden, die bereits gelesen wurden, und beschreibt nicht, wo und wie Daten gelesen werden sollen und wie und wo Daten geschrieben werden. Dies ist in den Quell- und Zielknoten in AWS Glue Studio konfiguriert. Weitere Informationen zu Rezepten finden Sie unter Rezepte erstellen und verwenden AWS Glue DataBrew.

Um AWS Glue DataBrew Rezepte in zu verwendenAWS Glue Studio, beginnen Sie mit dem Erstellen von Rezepten in AWS Glue DataBrew. Wenn Sie Rezepte haben, die Sie verwenden möchten, können Sie diesen Schritt überspringen.

IAM-Berechtigungen für AWS Glue DataBrew

Dieses Thema enthält Informationen zum besseren Verständnis der Aktionen und Ressourcen, die Sie als IAM-Administrator in einer AWS Identity and Access Management (IAM-) Richtlinie für die Transformation von Data Preparation Recipe verwenden können.

Weitere Informationen zur Sicherheit in finden Sie AWS Glue unter Access Management.

> **Note**
>
> In der folgenden Tabelle sind die Berechtigungen aufgeführt, die ein Benutzer benötigt, um ein vorhandenes AWS Glue DataBrew Rezept zu importieren.

Datenvorbereitung, Rezept, Transformationsaktionen

Aktion	Beschreibung
`databrew:ListRecipes`	Erteilt die Erlaubnis zum Abrufen von AWS Glue DataBrew Rezepten.
`databrew:ListRecipeVersions`	Erteilt die Erlaubnis zum Abrufen von AWS Glue DataBrew Rezeptversionen.
`databrew:DescribeRecipe`	Erteilt die Berechtigung zum Abrufen der AWS Glue DataBrew Rezeptbeschreibung.

Die Rolle, die Sie für den Zugriff auf diese Funktion verwenden, sollte über eine Richtlinie verfügen, die mehrere AWS Glue DataBrew Aktionen zulässt. Sie können dies erreichen, indem Sie entweder die `AWSGlueConsoleFullAccess` Richtlinie verwenden, die die erforderlichen Aktionen enthält, oder indem Sie Ihrer Rolle die folgende Inline-Richtlinie hinzufügen:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "databrew:ListRecipes",
                "databrew:ListRecipeVersions",
                "databrew:DescribeRecipe"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Um die Datenvorbereitungsrezept-Transformation zu verwenden, müssen Sie die `IAM:PassRole`-Aktion zur Berechtigungsrichtlinie hinzufügen.

Zusätzliche erforderliche Berechtigungen

Aktion	Beschreibung
`iam:PassRole`	Gewährt IAM die Berechtigung, dem Benutzer die Übergabe der genehmigten Rollen zu ermöglichen.

Ohne diese Berechtigungen tritt der folgende Fehler auf:

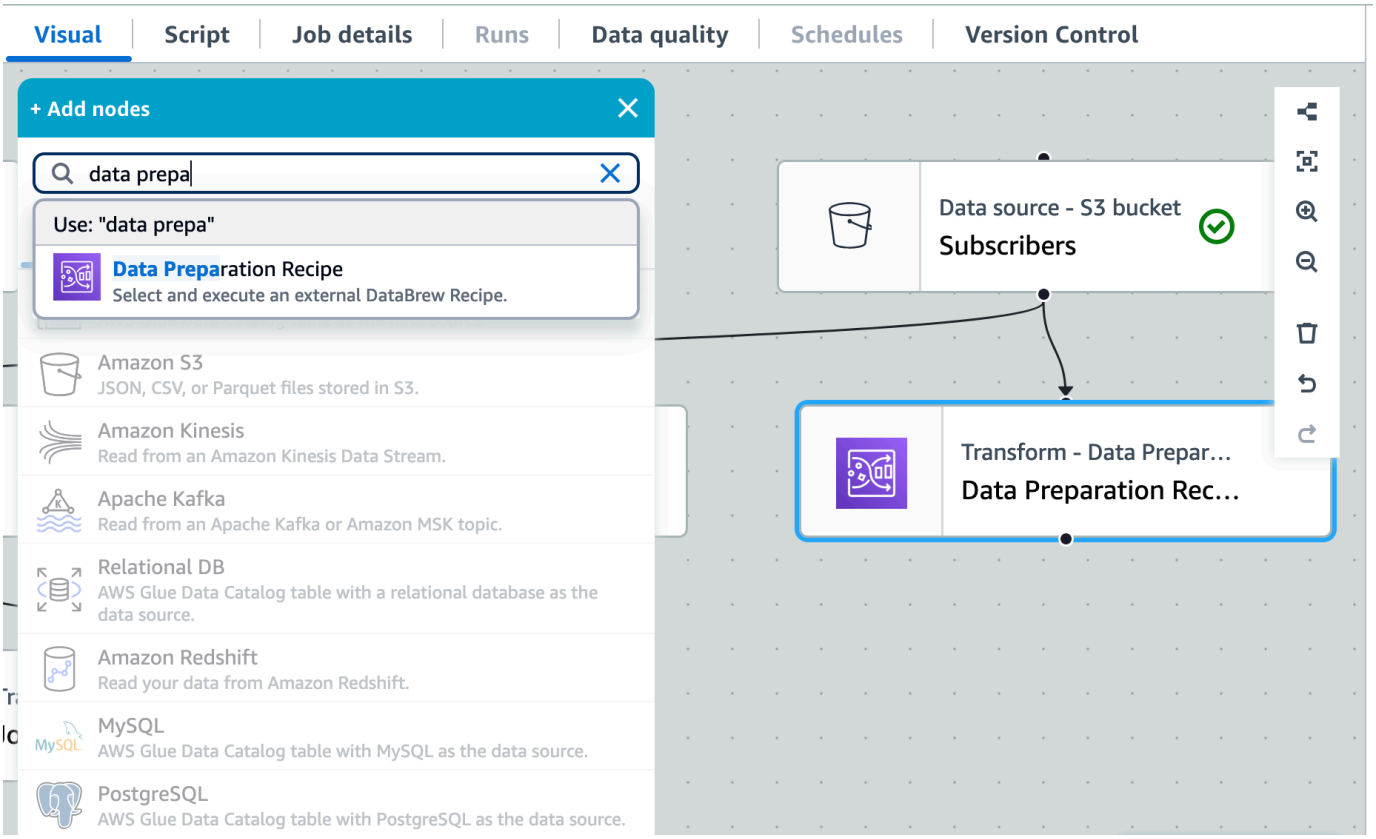
```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not
authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-
role/AWSGlueServiceRole
because no identity-based policy allows the iam:PassRole action"
```

Ein AWS Glue DataBrew Rezept importieren

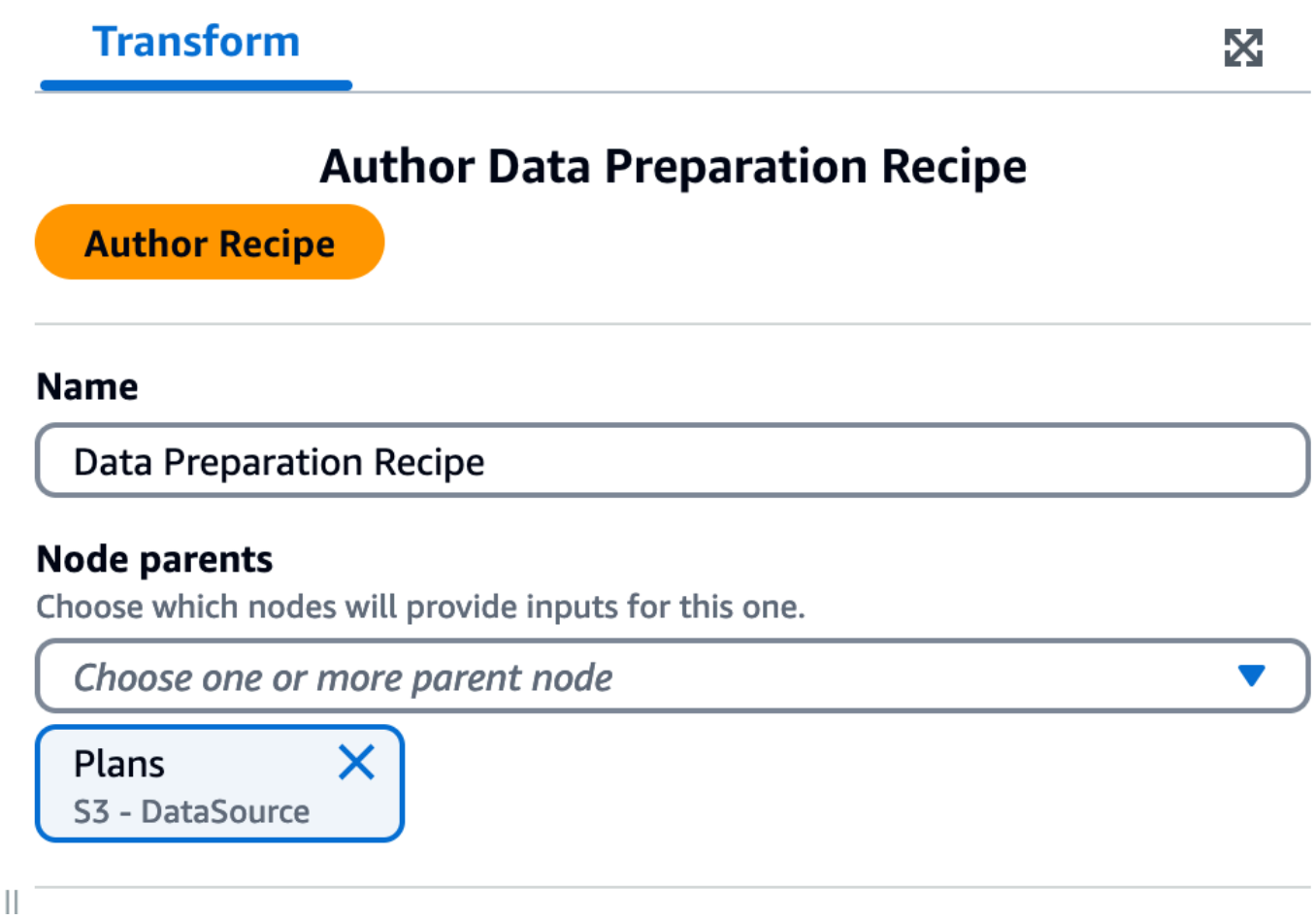
So importieren Sie ein AWS Glue DataBrew Rezept und verwenden es inAWS Glue Studio:

Wenn Sie bereits über einen Datenvorbereitungs-Rezeptknoten verfügen und die Rezeptschritte direkt darin bearbeiten möchten AWS Glue Studio, müssen Sie die Rezeptschritte in Ihren AWS Glue Studio Job importieren.

1. Starten Sie einen AWS Glue-Auftrag in AWS Glue Studio mit einer Datenquelle.
2. Fügen Sie den Knoten Data Preparation Recipe zur Job-Canvas hinzu.

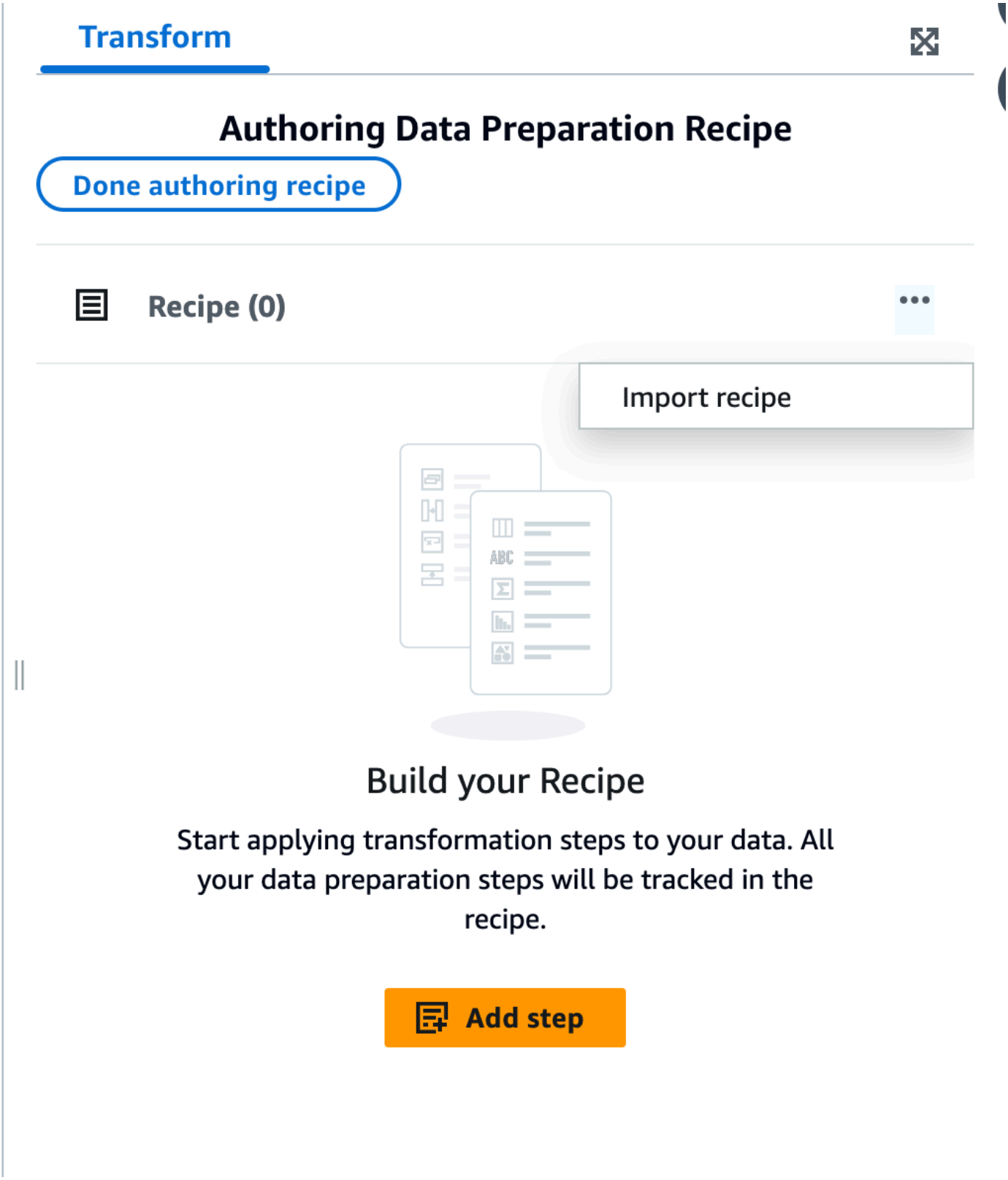


3. Geben Sie im Bedienfeld „Transformieren“ einen Namen für Ihr Rezept ein.
4. Wählen Sie einen oder mehrere übergeordnete Knoten aus, indem Sie die verfügbaren Knoten auf der Arbeitsfläche aus der Drop-down-Liste auswählen.
5. Wählen Sie „Autorenrezept“. Wenn die Option „Rezept erstellen“ grau ist, ist sie erst verfügbar, wenn die übergeordneten Knoten ausgewählt wurden und eine Datenvorschau-Sitzung abgeschlossen ist.

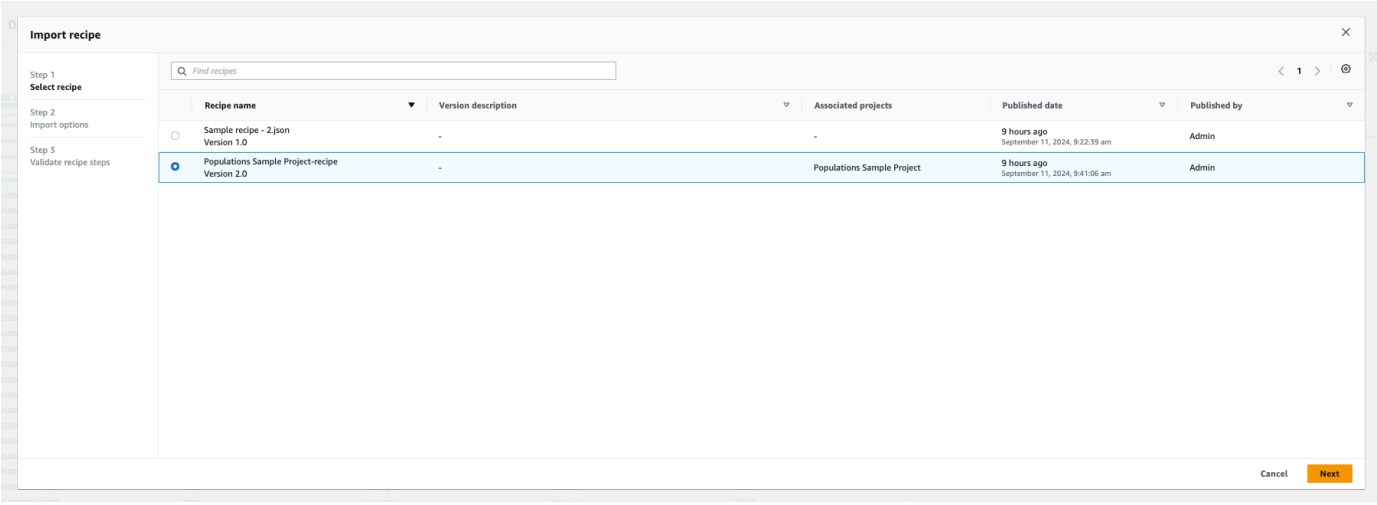


6. Der Datenrahmen wird geladen und zeigt Ihnen detaillierte Informationen zu Ihren Quelldaten an.

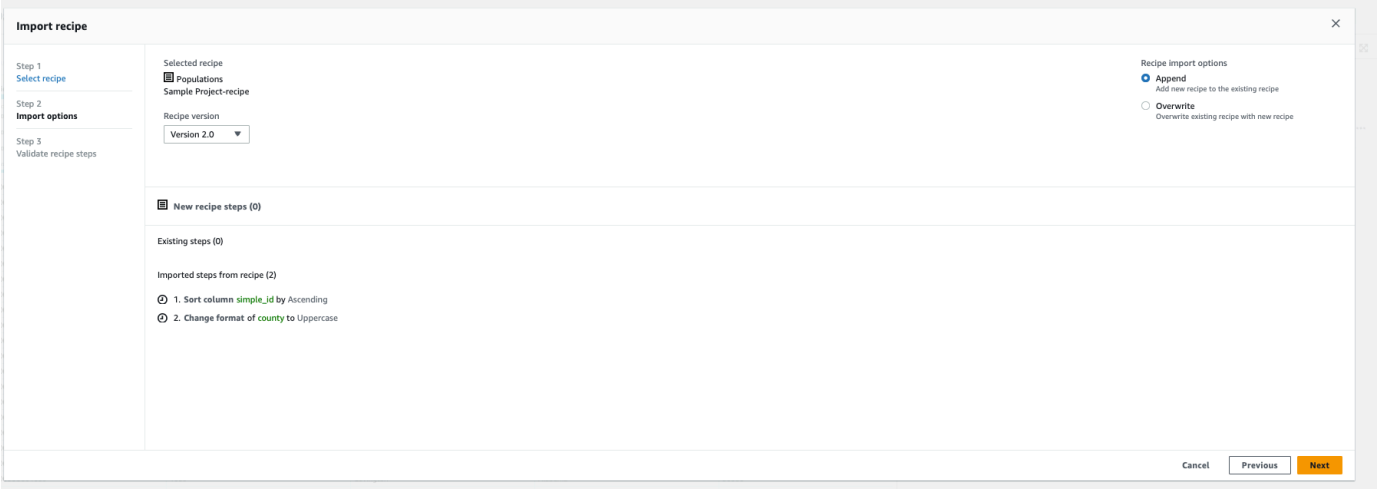
   Klicken Sie auf das Symbol „Weitere Aktionen“ und wählen Sie „Rezept importieren“.



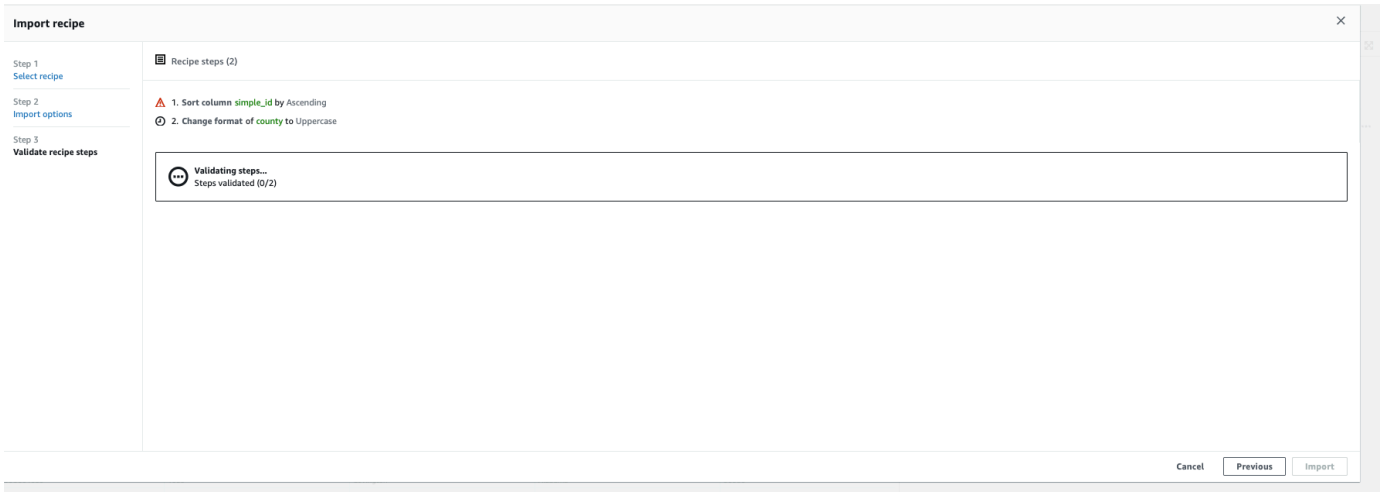
7. Verwenden Sie den Assistenten zum Importieren von Rezepten, um die Schritte abzuschließen. Suchen Sie in Schritt 1 nach Ihrem Rezept, wählen Sie es aus und klicken Sie auf Weiter.

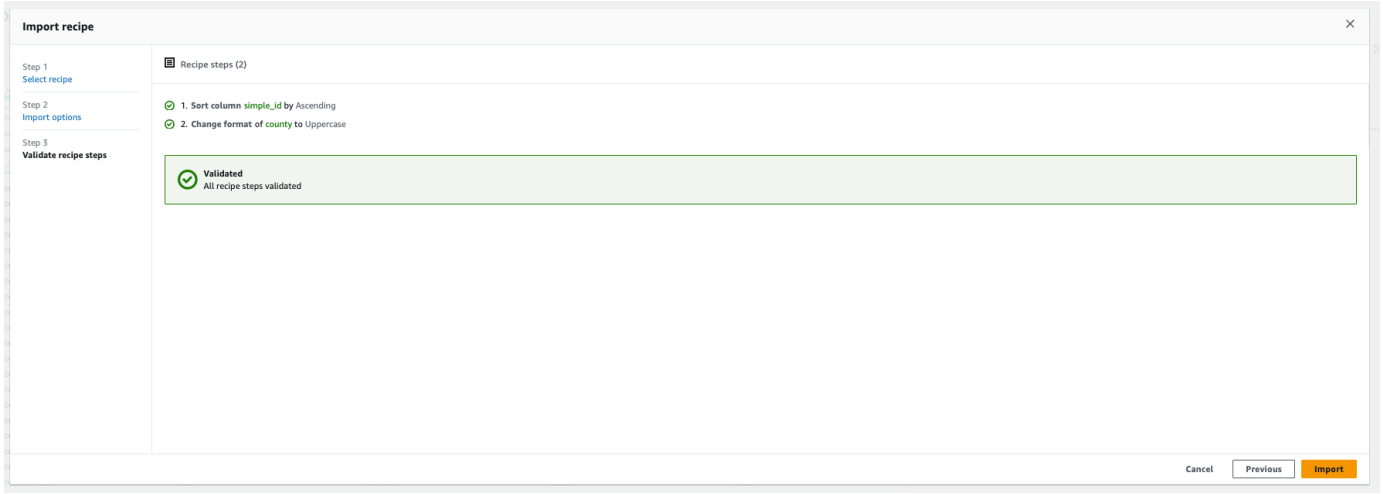


8. Wählen Sie in Schritt 2 Ihre Importoptionen aus. Sie können wählen, ob Sie ein neues Rezept an ein vorhandenes Rezept anhängen oder ein vorhandenes Rezept überschreiben möchten. Wählen Sie Weiter aus.



9. Validieren Sie in Schritt 3 die Rezeptschritte. Sobald Sie Ihr AWS Glue DataBrew Rezept importiert haben, können Sie dieses Rezept direkt in bearbeiten AWS Glue Studio.





10. Danach werden die Schritte als Teil Ihres AWS Glue Jobs importiert. Nehmen Sie auf der Registerkarte „Auftragsdetails“ die erforderlichen Konfigurationsänderungen vor, z. B. benennen Sie Ihren Job und passen Sie die zugewiesene Kapazität nach Bedarf an. Wählen Sie Speichern, um Ihren Job und Ihr Rezept zu speichern.

> **Note**
>
> JOIN, UNION, GROUP_BY, PIVOT, UNPIVOT, TRANSPOSE werden für den Rezeptimport nicht unterstützt und sind auch nicht im Rezepturerstellungsmodus verfügbar.

11. Optional können Sie die Erstellung des Jobs beenden, indem Sie nach Bedarf weitere Transformationsknoten und Datenzielknoten hinzufügen.

    Wenn Sie die Reihenfolge der Schritte nach dem Import eines Rezepts ändern, werden AWS Glue diese Schritte überprüft. Wenn Sie beispielsweise eine Spalte umbenannt und

anschließend gelöscht haben und den Löschschritt nach oben verschoben haben, wäre der Umbenennungsschritt ungültig. Anschließend können Sie die Schritte bearbeiten, um den Validierungsfehler zu beheben.

Migration von AWS Glue DataBrew zu AWS Glue Studio

Wenn Sie Rezepte gespeichert haben AWS Glue DataBrew, verwenden Sie die folgende Checkliste, um Ihre Rezepte nach AWS Glue Studio zu migrieren.

Wenn du möchtest	Dann tun Sie das
Erlauben Sie Benutzern das Abrufen von AWS Glue DataBrew Rezepten, Rezeptversionen und Rezeptbeschreibungen.	Fügen Sie IAM-Berechtigungen zu einer Richtlinie hinzu, die Ihrer Rolle den Zugriff auf die erforderlichen Aktionen ermöglicht. Siehe IAM-Berechtigungen für AWS Glue DataBrew.
Importiert ein vorhandenes AWS Glue DataBrew Rezept in AWS Glue Studio.	Führen Sie die Schritte unter Ein AWS Glue DataBrew Rezept importieren aus.
Importiert ein Rezept mit JOIN und UNION.	Rezepte mit UNION- und JOIN-Transformationen werden nicht unterstützt. Verwenden Sie die Transformationen Join and Union in AWS Glue Studio vor oder nach einem Data Preparation Recipe Node.

Verwenden von Schema ändern zum Neuzuordnen von Dateneigenschaftsschlüsseln

Eine Transformation für Schema ändern ordnet die Eigenschaftsschlüssel der Quelldaten den gewünschten Konfigurationen für die Zieldaten zu. In einem Transformationsknoten für Schema ändern können Sie:

- den Namen mehrerer Dateneigenschaftsschlüssel ändern;
- den Datentyp der Dateneigenschaftsschlüssel ändern, wenn der neue Datentyp unterstützt wird und zwischen den beiden Datentypen ein Transformationspfad vorhanden ist;
- eine Teilmenge von Dateneigenschaftsschlüsseln auswählen, indem Sie angeben, welche Dateneigenschaftsschlüssel gelöscht werden sollen.

Sie können dem Jobdiagramm bei Bedarf auch zusätzliche Change-Schema-Knoten hinzufügen, z. B. um zusätzliche Datenquellen zu ändern oder um einer Join-Transformation zu folgen.

Verwenden Sie „Schema ändern“ mit einem dezimalen Datentyp

Wenn Sie die Change-Schema-Transformation mit dem Dezimal-Datentyp verwenden, ändert die Change-Schema-Transformation die Genauigkeit auf den Standardwert (10,2). Um dies zu ändern und die Genauigkeit für Ihren Anwendungsfall festzulegen, können Sie die SQL-Query-Transformation verwenden und die Spalten mit einer bestimmten Genauigkeit umwandeln.

Wenn Sie beispielsweise eine Eingabespalte mit dem Namen "DecimalCol" vom Typ Dezimal haben und diese einer Ausgabespalte mit dem Namen "OutputDecimalCol" mit einer bestimmten Genauigkeit von (18,6) neu zuordnen möchten, gehen Sie wie folgt vor:

1. Fügen Sie nach der Transformation zum Ändern des Schemas eine nachfolgende SQL-Abfragetransformation hinzu.
2. Verwenden Sie in der SQL-Abfragetransformation eine SQL-Abfrage, um die neu zugeordnete Spalte mit der gewünschten Genauigkeit umzuwandeln. Die SQL-Abfrage würde wie folgt aussehen:

   ```
   SELECT col1, col2, CAST(DecimalCol AS DECIMAL(18,6)) AS OutputDecimalCol
   FROM __THIS__
   ```

   In der obigen SQL-Abfrage:

   - `col1` und `col2` sind andere Spalten in Ihren Daten, die Sie unverändert durchgehen möchten.
   - `DecimalCol` ist der ursprüngliche Spaltenname aus den Eingabedaten.
   - `CAST (DecimalCol AS DECIMAL (18,6)) `wandelt den Wert` DecimalCol `in einen Dezimaltyp mit einer Genauigkeit von 18 Ziffern und 6 Dezimalstellen um.
   - `AS OutputDecimalCol `benennt die gecastete Spalte in` `um. OutputDecimalCol

Mithilfe der SQL-Abfragetransformation können Sie die von der Change-Schema-Transformation festgelegte Standardgenauigkeit überschreiben und die Dezimalspalten explizit auf die gewünschte Genauigkeit umwandeln. Dieser Ansatz ermöglicht es Ihnen, die Change-Schema-Transformation für das Umbenennen und Restrukturieren Ihrer Daten zu nutzen und gleichzeitig die Genauigkeitsanforderungen für Dezimalspalten bei der nachfolgenden SQL-Query-Transformation zu erfüllen.

Hinzufügen einer Change-Schema-Transformation zu Ihrem Job

> **Note**
>
> Bei der Transformation für Schema ändern wird nicht zwischen Groß- und Kleinschreibung unterschieden.

So fügen Sie Ihrem Auftragsdiagramm einen Knoten für Schema ändern hinzu

1. (Optional) Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Schema ändern aus, um Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzuzufügen.
2. Geben Sie im Bereich Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transformation im Bedienfeld Knoteneigenschaften.
4. Ändern Sie das Eingabeschema:
   - Um einen Dateneigenschaftsschlüssel umzubenennen, geben Sie den neuen Namen in das Feld Target key (Zielschlüssel) ein.
   - Um den Datentyp eines Dateneigenschaftsschlüssels zu ändern, wählen Sie den neuen Datentyp für den Schlüssel aus der Liste Data type (Datentyp) aus.
   - Um einen Dateneigenschaftsschlüssel aus dem Zielschema zu entfernen, aktivieren Sie das Kästchen Drop (Auslassen) beim jeweiligen Schlüssel.
5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-

Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Nutzung von Duplikate verwerfen

Die Transformation Duplikate verwerfen entfernt Zeilen aus Ihrer Datenquelle und bietet Ihnen zwei Optionen. Sie können die doppelten Zeilen entfernen, die völlig identisch sind, oder Sie können die Felder auswählen, die übereinstimmen sollen, und nur die Zeilen entfernen, die auf den von Ihnen ausgewählten Feldern basieren.

In diesem Datensatz haben Sie zum Beispiel doppelte Zeilen, bei denen alle Werte in einigen Zeilen genau gleich sind wie in einer anderen Zeile, und einige der Werte in den Zeilen sind gleich oder unterschiedlich.

Zeile	Name	Email	Age	Status	Hinweis
1	Joy	joy@gmail	33	NY	
2	Tim	tim@gmail	45	OH	
3	Rose	rose@gmail	23	NJ	
4	Tim	tim@gmail	42	OH	
5	Rose	rose@gmail	23	NJ	
6	Tim	tim@gmail	42	OH	dies ist eine doppelte Zeile und entspricht in allen Werten vollständig der Zeile Nr. 4
7	Rose	rose@gmail	23	NJ	Dies ist eine doppelte Zeile und entspricht in

Zeile	Name	Email	Age	Status	Hinweis
					allen Werten vollständig der Zeile Nr. 5

Wenn Sie sich dafür entscheiden, ganze Zeilen abzugleichen, werden die Zeilen 6 und 7 aus dem Datensatz entfernt. Der Datensatz lautet nun:

Zeile	Name	Email	Age	Status
1	Joy	joy@gmail	33	NY
2	Tim	tim@gmail	45	OH
3	Rose	rose@gmail	23	NJ
4	Tim	tim@gmail	42	OH
5	Rose	rose@gmail	23	NJ

Wenn Sie Schlüssel angeben möchten, können Sie Zeilen entfernen, die mit „Name“ und „E-Mail“ übereinstimmen. Auf diese Weise können Sie genauer festlegen, was eine „doppelte Zeile“ für Ihren Datensatz ist. Durch Angabe von „Name“ und „E-Mail“ lautet der Datensatz nun:

Zeile	Name	Email	Age	Status
1	Joy	joy@gmail	33	NY
2	Tim	tim@gmail	45	OH
3	Rose	rose@gmail	23	NJ

Einige Dinge, die Sie berücksichtigen sollten:

- Damit Zeilen als Duplikat erkannt werden, muss bei Werten die Groß- und Kleinschreibung beachtet werden. Alle Werte in Zeilen müssen die gleiche Groß- und Kleinschreibung haben – dies gilt für beide von Ihnen gewählten Optionen („Ganze Zeilen abgleichen“ oder „Schlüssel angeben“).
- Alle Werte werden als Zeichenfolgen eingelesen.
- Die Transformation Duplikate verwerfen verwendet den Spark-Befehl dropDuplicates.
- Wenn Sie die Transformation Duplikate verwerfen verwenden, wird die erste Zeile beibehalten und die anderen Zeilen werden gelöscht.
- Die Transformation Duplikate verwerfen ändert das Schema des Datenrahmens nicht. Wenn Sie die Angabe von Schlüsseln auswählen, werden alle Felder in dem resultierenden Datenrahmen beibehalten.

Wird verwendet SelectFields , um die meisten Dateneigenschaftsschlüssel zu entfernen

Mithilfe der SelectFieldsTransformation können Sie eine Teilmenge von Dateneigenschaftsschlüsseln aus der Datenmenge erstellen. Dabei geben Sie an, welche Dateneigenschaftsschlüssel beibehalten werden sollen. Der Rest wird dann aus dem Datensatz entfernt.

> **Note**
>
> Bei der SelectFieldsTransformation wird zwischen Groß- und Kleinschreibung unterschieden. Verwenden Sie diese Option, ApplyMappingwenn Sie Felder ohne Berücksichtigung der Groß- und Kleinschreibung auswählen möchten.

Um Ihrem Jobdiagramm einen SelectFields Transformationsknoten hinzuzufügen

1. (Optional) Öffnen Sie das Bedienfeld „Ressourcen“ und wählen Sie dann aus SelectFields, ob Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzufügen möchten.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) im Bereich mit den Knotendetails aus.

4. Wählen Sie unter der Überschrift SelectFieldsdie Dateneigenschaftsschlüssel im Datensatz aus, die Sie behalten möchten. Alle nicht ausgewählten Dateneigenschaftsschlüssel werden aus dem Datensatz gelöscht.

   Sie können auch das Kontrollkästchen neben der Spaltenüberschrift Field (Feld) aktivieren, um automatisch alle Dateneigenschaftsschlüssel im Datensatz auszuwählen. Anschließend können Sie die Auswahl einzelner Dateneigenschaftsschlüssel aufheben, damit sie aus dem Datensatz entfernt werden.

5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.

6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Wird verwendet DropFields , um die meisten Dateneigenschaftsschlüssel zu behalten

Mithilfe der DropFieldsTransformation können Sie aus dem Datensatz eine Teilmenge von Dateneigenschaftsschlüsseln erstellen. Dabei geben Sie an, welche Dateneigenschaftsschlüssel aus dem Datensatz gelöscht werden sollen. Die restlichen Schlüssel werden dann beibehalten.

> **Note**
>
> Bei der DropFieldsTransformation wird zwischen Groß- und Kleinschreibung unterschieden. Verwenden Sie Schema ändern, wenn bei der Auswahl von Feldern die Groß- und Kleinschreibung nicht berücksichtigt werden soll.

Um Ihrem Jobdiagramm einen DropFields Transformationsknoten hinzuzufügen

1. (Optional) Öffnen Sie das Bedienfeld „Ressourcen“ und wählen DropFieldsSie dann aus, ob Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzufügen möchten.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) im Bereich mit den Knotendetails aus.
4. Wählen Sie unter der Überschrift DropFieldsdie Dateneigenschaftsschlüssel aus, die Sie aus der Datenquelle löschen möchten.

   Sie können auch das Kontrollkästchen neben der Spaltenüberschrift Field (Feld) aktivieren, um automatisch alle Dateneigenschaftsschlüssel im Datensatz auszuwählen. Anschließend können Sie die Auswahl einzelner Dateneigenschaftsschlüssel aufheben, damit sie im Datensatz beibehalten werden.
5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Umbenennen eines Felds im Datensatz

Sie können die RenameFieldTransformation verwenden, um den Namen für einen einzelnen Eigenschaftsschlüssel im Datensatz zu ändern.

> **Note**
>
> Bei der RenameFieldTransformation wird zwischen Groß- und Kleinschreibung unterschieden. Verwenden Sie diese ApplyMappingOption, wenn Sie eine Transformation benötigen, bei der Groß-/Kleinschreibung nicht berücksichtigt wird.

> **Tip**
>
> Mit der Transformation für Schema ändern können Sie mehrere Dateneigenschaftsschlüssel im Datensatz mit einer einzigen Transformation umbenennen.

Um Ihrem Jobdiagramm einen RenameField Transformationsknoten hinzuzufügen

1. (Optional) Öffnen Sie das Bedienfeld „Ressourcen“ und wählen RenameFieldSie dann aus, ob Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzufügen möchten.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) aus.
4. Wählen Sie unter der Überschrift Data field (Datenfeld) einen Eigenschaftsschlüssel aus den Quelldaten aus und geben Sie dann einen neuen Namen in das Feld New field name (Neuer Feldname) ein.
5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-

Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Erstellen eines Datensatzbeispiels mit Spigot

Um zu überprüfen, ob die von Ihrem Auftrag durchgeführten Transformationen wie beabsichtigt funktionieren, könnten Sie ein Beispiel der Daten abrufen. Die Spigot-Transformation schreibt eine Teilmenge von Akten aus dem Datensatz in eine JSON-Datei in einem Amazon-S3-Bucket. Die Methode zum Datensampling kann entweder eine bestimmte Anzahl von Akten vom Anfang der Datei oder ein Wahrscheinlichkeitsfaktor sein, mit dem Akten ausgewählt werden.

Dem Auftragsdiagramm einen Spigot-Transformationsknoten hinzufügen

1. (Optional) Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Spigot aus, um Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzuzufügen.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) im Bereich mit den Knotendetails aus.
4. Geben Sie einen Amazon-S3-Pfad ein oder wählen Sie Browse S3 (S3 durchsuchen) aus, um einen Speicherort in Amazon S3 auszuwählen. Dies ist der Speicherort, an dem der Auftrag die JSON-Datei schreibt, die das Datenbeispiel enthält.
5. Geben Sie die Informationen für die Samplingmethode ein. Sie können einen Wert für Number of records (Anzahl der Akten) angeben, um ab dem Anfang des Datensatzes zu schreiben, sowie eine Probability threshold (Wahrscheinlichkeitsschwelle) für die Auswahl eines bestimmten Datensatzes (als Dezimalwert mit einem Maximalwert von 1).

   Um beispielsweise die ersten 50 Akten aus dem Datensatz zu schreiben, legen Sie Number of records (Anzahl der Datensätze) auf 50 fest und die Probability threshold (Wahrscheinlichkeitsschwelle) auf 1 (100 %).

Schaffen eines Verbunds von Datensätzen

Mit Join (Verbund) können Sie zwei Datensätze zu einem kombinieren. Hierbei geben Sie die Schlüsselnamen im Schema der einzelnen zu vergleichenden Datensätze an. Die Ausgabe (`DynamicFrame`) enthält Zeilen, in denen die Schlüssel die Bedingung für den Verbund erfüllen. Die

Zeilen in den Datensätzen, die diese Bedingung erfüllen, werden in der Ausgabe (`DynamicFrame`) zu einer einzigen Zeile zusammengefasst. Die Ausgabe enthält sämtliche Spalten, die in den Datensätzen vorkommen.

Dem Auftragsdiagramm einen Join-Transformationsknoten hinzufügen

1. Wenn nur eine Datenquelle verfügbar ist, müssen Sie dem Auftragsdiagramm einen neuen Datenquellknoten hinzufügen.
2. Wählen Sie einen der Quellknoten für die Zusammenführung aus. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Zusammenführen aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen.
3. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein.
4. Fügen Sie auf der Registerkarte Node properties (Knoteneigenschaften) unter der Überschrift Node parents (Übergeordnete Knoten) einen übergeordneten Knoten hinzu, sodass zwei Datensätze als Eingabe bereitstehen. Der übergeordnete Knoten kann ein Datenquellknoten oder ein Transformierungsknoten sein.

   > **Note**
   >
   > Ein Verbund kann nur zwei übergeordnete Knoten haben.

5. Wählen Sie die Registerkarte Transform (Transformation) aus.

   Wenn Sie eine Meldung sehen, dass Schlüsselnamen in Konflikt stehen, können Sie Folgendes tun:

   - Wählen Sie Resolve it, um Ihrem Jobdiagramm automatisch einen ApplyMappingTransformationsknoten hinzuzufügen. Der ApplyMapping Knoten fügt allen Schlüsseln im Datensatz, die denselben Namen wie ein Schlüssel im anderen Datensatz haben, ein Präfix hinzu. Mit dem Standardwert **`right`** werden zum Beispiel alle Schlüssel im rechten Datensatz, die denselben Namen wie ein Schlüssel im linken Datensatz haben, zu `(right)`*`key name`* umbenannt.
   - Fügen Sie zuvor im Auftragsdiagramm einen Transformationsknoten manuell hinzu, um die in Konflikt stehenden Schlüssel zu entfernen oder umzubenennen.

6. Wählen Sie den Join-Typ aus der Liste Join type (Verbundtyp) aus.

- Inner join: Gibt eine Zeile mit Spalten aus beiden Datensätze zurück, für jeden Treffer auf Grundlage der Bedingung für den Join. Zeilen, die die Bedingung nicht erfüllen, werden nicht zurückgegeben.
- Left join: Alle Zeilen aus dem linken Datensatz; die Zeilen aus dem rechten Datensatz nur, wenn sie die Bedingung für den Join erfüllen.
- Right join: Alle Zeilen aus dem rechten Datensatz; die Zeilen aus dem linken Datensatz nur, wenn sie die Bedingung für den Join erfüllen.
- Outer join: Alle Zeilen aus beiden Datensätzen.
- Left semi join: Alle Zeilen aus dem linken Datensatz, die eine Übereinstimmung im rechten Datensatz basierend auf der Join-Bedingung haben.
- Left anti join: Alle Zeilen im linken Datensatz, die keine Übereinstimmung im rechten Datensatz basierend auf der Join-Bedingung haben.

7. Wählen Sie auf der Registerkarte Transform (Transformation) unter der Überschrift Join conditions (Join-Bedingungen) die Option Add condition (Bedingung hinzufügen) aus. Wählen Sie einen Eigenschaftsschlüssel aus jedem zu vergleichenden Datensatz aus. Eigenschaftsschlüssel links vom Vergleichsoperator werden als „linker Datensatz“ bezeichnet, Eigenschaftsschlüssel recht davon als „rechter Datensatz“.

   Für komplexere Join-Bedingungen können Sie zusätzliche passende Schlüssel hinzufügen, indem Sie Add condition (Bedingung hinzufügen) öfter als einmal auswählen. Wenn Sie versehentlich eine Bedingung hinzufügen, können Sie sie mit dem Lösch-Symbol ( ) wieder entfernen.

8. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
9. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-

Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Ein Beispiel für das Join-Ausgabeschema wäre eine Verknüpfung zwischen zwei Datensätzen mit den folgenden Eigenschaftenschlüsseln:

```
Left: {id, dept, hire_date, salary, employment_status}
Right: {id, first_name, last_name, hire_date, title}
```

Der Join ist so konfiguriert, dass es bei den Schlüsseln `id` und `hire_date` mit dem Vergleichsoperator = zu einem Treffer kommt.

Da beide Datensätze die Schlüssel `id` und `hire_date` enthalten, haben Sie die Option Resolve it (Problem lösen) ausgewählt, um den Schlüsseln im rechten Datensatz automatisch das Präfix **`right`** anzuhängen.

Die Schlüssel im Ausgabeschema wären:

```
{id, dept, hire_date, salary, employment_status,
(right)id, first_name, last_name, (right)hire_date, title}
```

Verwendung von Union zum Kombinieren von Zeilen

Sie verwenden den Union-Transformationsknoten, wenn Sie Zeilen aus mehr als einer Datenquelle kombinieren möchten, die dasselbe Schema haben.

Es gibt zwei Arten von Union-Transformationen:

1. ALL – Wenn ALL angewendet wird, werden durch die resultierende Vereinigung keine doppelten Zeilen entfernt.
2. DISTINCT – Bei der Anwendung von DISTINCT entfernt die resultierende Vereinigung doppelte Zeilen.

Vereinigungen im Vergleich zu Zusammenführungen

Sie verwenden Union, um Zeilen zu kombinieren. Sie verwenden Join, um Spalten zu kombinieren.

Verwendung der Union-Transformation im Visual-ETL-Zeichenbereich

1. Fügen Sie mehr als eine Datenquelle hinzu, um eine Vereinigungstransformation durchzuführen. Um eine Datenquelle hinzuzufügen, öffnen Sie das Bedienfeld Ressourcen und wählen Sie die Datenquelle auf der Registerkarte Quellen aus. Bevor Sie die Union-Transformation verwenden, müssen Sie sicherstellen, dass alle an der Vereinigung beteiligten Datenquellen dasselbe Schema und dieselbe Struktur haben.
2. Wenn Sie über mindestens zwei Datenquellen verfügen, die Sie mithilfe der Union-Transformation kombinieren möchten, erstellen Sie die Union-Transformation, indem Sie sie dem Zeichenbereich hinzufügen. Öffnen Sie im Zeichenbereich das Ressourcenfenster und suchen Sie nach „Union“. Sie können auch die Registerkarte Transformationen im Ressourcenfenster auswählen und nach unten scrollen, bis Sie die Union-Transformation finden, und dann Union auswählen.
3. Wählen Sie den Knoten Union im Zeichenbereich des Auftrags aus. Wählen Sie im Fenster Knoteneigenschaften die übergeordneten Knoten aus, die mit der Union-Transformation verbunden werden sollen.
4. AWS Glue prüft die Kompatibilität, um sicherzustellen, dass die Unionstransformation auf alle Datenquellen angewendet werden kann. Wenn das Schema für die Datenquellen identisch ist, ist der Vorgang zulässig. Wenn die Datenquellen nicht dasselbe Schema haben, wird eine ungültige Fehlermeldung angezeigt: „Die Eingabeschemas dieser Verbindung sind nicht identisch.“ Erwägen Sie ApplyMapping die Verwendung, um die Schemas abzugleichen.“ Um dieses Problem zu beheben, wählen Sie „Verwenden ApplyMapping“.
5. Wählen Sie den Union-Typ.

    1. Alle – Standardmäßig ist der Typ „All Union“ ausgewählt. Dies führt zu doppelten Zeilen, sofern in der Datenkombination solche vorhanden sind.
    2. Eindeutig – Wählen Sie „Eindeutig“, wenn doppelte Zeilen aus der resultierenden Datenkombination entfernt werden sollen.

Wird verwendet SplitFields , um einen Datensatz in zwei Teile aufzuteilen

Die SplitFieldsTransformation ermöglicht es Ihnen, einige der Dateneigenschaftsschlüssel im Eingabe-Dataset auszuwählen und sie in einem Datensatz und die nicht ausgewählten Schlüssel in einem separaten Datensatz zusammenzufassen. Als Ausgabe entstehen bei dieser Transformation eine Reihe von `DynamicFrames`.

> **Note**
>
> Sie müssen eine SelectFromCollectionTransformation verwenden, um die Sammlung von `DynamicFrames` in eine einzelne zu konvertieren, `DynamicFrame` bevor Sie die Ausgabe an einen Zielort senden können.

Bei der SplitFieldsTransformation wird zwischen Groß- und Kleinschreibung unterschieden. Fügen Sie eine ApplyMappingTransformation als übergeordneten Knoten hinzu, wenn Sie Eigenschaftsschlüsselnamen benötigen, bei denen Groß- und Kleinschreibung nicht beachtet werden muss.

Um Ihrem Jobdiagramm einen SplitFields Transformationsknoten hinzuzufügen

1. (Optional) Öffnen Sie das Bedienfeld „Ressourcen“ und wählen SplitFieldsSie dann aus, ob Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzufügen möchten.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) aus.
4. Wählen Sie aus, welche Eigenschaftsschlüssel Sie in den ersten Datensatz aufnehmen möchten. Die Schlüssel, die Sie nicht auswählen, kommen in den zweiten Datensatz.
5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

7. Konfigurieren Sie einen SelectFromCollectionTransformationsknoten, um die resultierenden Datensätze zu verarbeiten.

Übersicht über die SelectFromCollection-Transformation

Bestimmte Transformationen haben mehrere Datasets als Ausgabe anstelle eines einzelnen Datensatzes, zum Beispiel. SplitFields Die SelectFromCollectionTransformation wählt einen Datensatz (`DynamicFrame`) aus einer Sammlung von Datensätzen (ein Array von) aus. `DynamicFrames` Die Ausgabe für die Transformation ist der ausgewählte `DynamicFrame`.

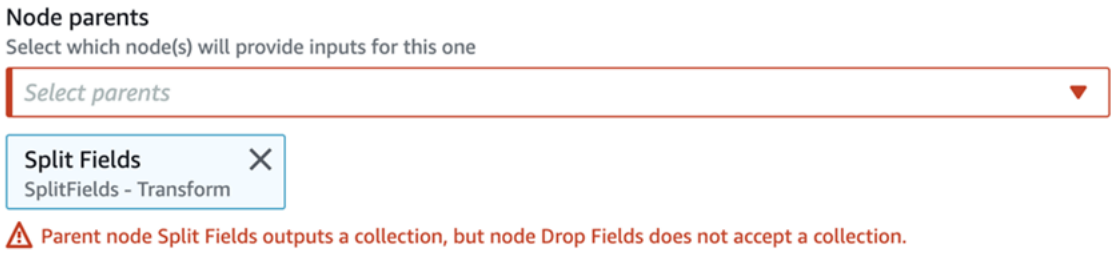
Diese Transformation ist gefragt, wenn Sie zuvor eine Transformation verwendet haben, bei der Sammlung von `DynamicFrames` entsteht, zum Beispiel:

- Transformationen für benutzerdefinierten Code
- SplitFields

Wenn Sie Ihrem Auftragsdiagramm nach einer dieser Transformationen keinen SelectFromCollectionTransformationsknoten hinzufügen, erhalten Sie eine Fehlermeldung für Ihren Job.

Der übergeordnete Knoten für diese Transformation muss ein Knoten sein, der eine Sammlung von `DynamicFrames`zurückgibt. Wenn Sie einen übergeordneten Knoten für diesen Transformationsknoten auswählen, der einen einzelnen `DynamicFrame` zurückgibt, z. B. eine Join-Transformation, gibt der Auftrag einen Fehler zurück.

Ebenso gibt Ihr Job einen Fehler zurück, wenn Sie einen SelectFromCollectionKnoten in Ihrem Job-Diagramm als übergeordneten Knoten für eine Transformation verwenden, die eine Single `DynamicFrame` als Eingabe erwartet.



Wird verwendet SelectFromCollection , um auszuwählen, welcher Datensatz beibehalten werden soll

Verwenden Sie die SelectFromCollectionTransformation, um eine Sammlung von `DynamicFrames` in eine einzelne zu konvertieren`DynamicFrame`.

Um Ihrem Jobdiagramm einen SelectFromCollection Transformationsknoten hinzuzufügen

1. (Optional) Öffnen Sie das Bedienfeld „Ressourcen“ und wählen SelectFromCollectionSie dann aus, ob Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzufügen möchten.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) aus.
4. Wählen Sie unter der Überschrift Frame index (Frame-Index) die Array-Indexnummer aus, die zum `DynamicFrame` gehört, den Sie aus der Sammlung von `DynamicFrames` auswählen möchten.

   Wenn es sich bei dem übergeordneten Knoten für diese Transformation beispielsweise um eine SplitFieldsTransformation handelt, können Sie auf der Registerkarte Ausgabeschema dieses Knotens das Schema für jeden Knoten sehen`DynamicFrame`. Wenn Sie möchten, dass der `DynamicFrame` weiterhin mit dem Schema für Output 2 (Ausgabe 2) verknüpft ist, wählen Sie für Frame index (Frame-Index) den Wert **1**, der der zweite Wert in der Liste ist.

   Nur der von Ihnen gewählte `DynamicFrame` wird in die Ausgabe einbezogen.
5. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-

Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Suchen und Ausfüllen fehlender Werte in einem Datensatz

Sie können die FillMissingValuesTransformation verwenden, um Datensätze in der Datenmenge zu finden, in denen Werte fehlen, und ein neues Feld mit einem durch Imputation bestimmten Wert hinzuzufügen. Mit dem Eingabedatensatz wird das Modell zum Machine Learning (ML) trainiert, das den fehlenden Wert bestimmt. Wenn Sie inkrementelle Datensätze verwenden, wird jeder inkrementelle Satz als Trainingsdaten für das ML-Modell verwendet, sodass die Ergebnisse möglicherweise nicht so genau sind.

Um einen FillMissingValues Transformationsknoten in Ihrem Jobdiagramm zu verwenden

1. (Optional) Öffnen Sie FillMissingValuesdas Bedienfeld „Ressourcen“ und fügen Sie Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzu.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie die Registerkarte Transform (Transformation) aus.
4. Wählen Sie bei Data field (Datenfeld) den Namen der Spalte bzw. des Felds aus den Quelldaten aus, die/das auf fehlende Werte hin analysiert werden sollen.
5. (Optional) Geben Sie im Feld New field name (Neuer Feldname) einen Namen für das Feld ein, das den einzelnen Akten hinzugefügt wird und den geschätzten Ersatzwert für das analysierte Feld enthält. Wenn das analysierte Feld keinen fehlenden Wert hat, wird der Wert im analysierten Feld in das neue Feld kopiert.

   Falls Sie keinen Namen für das neue Feld angeben, ist der Standardname der Name der analysierten Spalte plus dem Anhang `_filled`. Wenn Sie z. B. **`Age`** bei Data field (Datenfeld) eingeben und keinen Wert für New field name (Neuer Feldname) angeben, erhalten alle Akten ein neues Feld mit dem Namen **`Age_filled`**.
6. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-

Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.

7. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Filtern von Schlüsseln in einem Datensatz

Erstellen Sie einen neuen Datensatz mithilfe von Filtern, durch die Akten aus dem Eingabe-Datensatz basierend auf einem regulären Ausdruck gefiltert werden. Zeilen, die die Filterbedingung nicht erfüllen, kommen nicht in die Ausgabe.

- Bei String-Datentypen können Sie Zeilen herausfiltern, in denen der Schlüsselwert einer angegebenen Zeichenfolge entspricht.
- Bei numerischen Datentypen können Sie Zeilen herausfiltern, indem ein angegebener Wert mit dem Schlüsselwert anhand der Vergleichsoperatoren <, >, =, !=, <= und >= verglichen wird.

Wenn Sie mehrere Filterbedingungen angeben, gilt für die Ergebnisse standardmäßig der Operator AND, sie können aber auch OR anwenden lassen.

Bei der Filter-Transformation wird die Groß- und Kleinschreibung berücksichtigt. Fügen Sie eine ApplyMappingTransformation als übergeordneten Knoten hinzu, wenn Sie Eigenschaftsschlüsselnamen benötigen, bei denen Groß- und Kleinschreibung nicht beachtet werden muss.

Dem Auftragsdiagramm einen Filter-Transformationsknoten hinzufügen

1. (Optional) Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Filter aus, um Ihrem Auftragsdiagramm bei Bedarf eine neue Transformation hinzuzufügen.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.

3. Wählen Sie die Registerkarte Transform (Transformation) aus.
4. Wählen Sie entweder Global AND oder Global OR aus. Das bestimmt, was bei mehreren Filterbedingungen geschieht. Alle Bedingungen unterliegen entweder dem Operator `AND` oder `OR`. Wenn Sie nur eine einzelne Filterbedingungen haben, können Sie irgend einen auswählen.
5. Fügen Sie über die Schaltfläche Add condition (Bedingung hinzufügen) im Abschnitt Filter condition (Filterbedingung) eine Filterbedingung hinzu.

   Wählen Sie im Feld Key (Schlüssel) einen Eigenschaftsschlüsselnamen aus dem Datensatz aus. Wählen Sie im Feld Operation den Vergleichsoperator aus. Geben Sie im Feld Value (Wert) den Vergleichswert ein. Hier sind einige Beispiele für Filterbedingungen:

   - `year >= 2018`
   - `State matches 'CA*'`

   Wenn Sie nach Strings filtern, stellen Sie sicher, dass der Vergleichswert ein Format für reguläre Ausdrücke hat, das der in den Auftragseigenschaften ausgewählten Skriptsprache entspricht (Python oder Scala).
6. Geben Sie nach Bedarf zusätzliche Filterbedingungen an.
7. (Optional) Nachdem Sie die Eigenschaften des Transformationsknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das geänderte Schema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
8. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

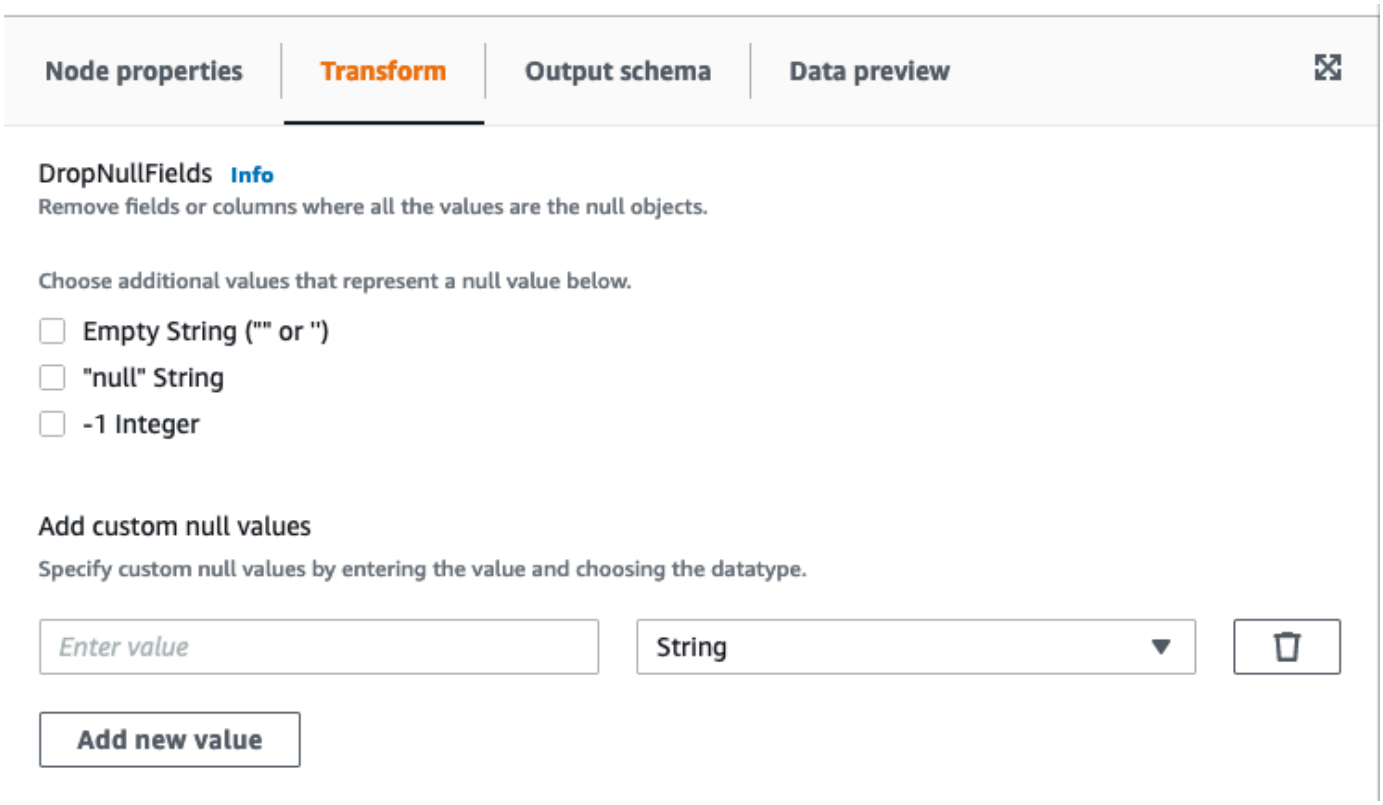
Wird verwendet DropNullFields , um Felder mit Nullwerten zu entfernen

Verwenden Sie die DropNullFieldsTransformation, um Felder aus dem Datensatz zu entfernen, wenn alle Werte im Feld „Null“ sind. Standardmäßig AWS Glue Studio erkennt Null-Objekte, aber

einige Werte wie leere Zeichenketten, Zeichenketten, die „Null“ sind, ganze Zahlen -1 oder andere Platzhalter wie Nullen, werden nicht automatisch als Nullen erkannt.

Um das zu verwenden DropNullFields

1. Fügt dem Jobdiagramm einen DropNullFields Knoten hinzu.
2. Auf der Registerkarte Knoteneigenschaften wählen Sie zusätzliche Werte aus, die einen Nullwert darstellen. Sie können wählen, ob Sie keinen oder alle Werte auswählen möchten:



   - Leerer String ("" oder '') - Felder, die leere Strings enthalten, werden entfernt
   - „null string“ - Felder, die die Zeichenfolge mit dem Wort 'null' enthalten, werden entfernt
   - -1 integer - Felder, die eine Ganzzahl von -1 (negativ) enthalten, werden entfernt
3. Bei Bedarf können Sie auch benutzerdefinierte Nullwerte angeben. Dies sind Nullwerte, die für Ihren Datensatz eindeutig sein können. Um einen benutzerdefinierten Nullwert hinzuzufügen, wählen Sie Add new value (Neuen Wert hinzufügen).
4. Geben Sie den benutzerdefinierten Null-Wert ein. Dies kann beispielsweise Null oder ein beliebiger Wert sein, der verwendet wird, um eine Null im Datensatz darzustellen.
5. Wählen Sie den Datentyp im Dropdown-Feld aus. Datentypen können entweder String oder Integer sein.

> **Note**
>
> Benutzerdefinierte Nullwerte und ihre Datentypen müssen genau übereinstimmen, damit die Felder als Nullwerte erkannt und die Felder entfernt werden. Teilweise Übereinstimmungen, bei denen nur der benutzerdefinierte Nullwert, nicht aber der Datentyp übereinstimmt, führen nicht dazu, dass die Felder entfernt werden.

Verwenden einer SQL-Abfrage zum Transformieren von Daten

Sie können mit einer SQL-Transformation Ihre eigene Transformation in Form einer SQL-Abfrage schreiben.

Ein SQL-Transformationsknoten kann mehrere Datensätze als Eingaben enthalten, erzeugt jedoch nur einen einzigen Datensatz als Ausgabe. Enthalten ist ein Textfeld, in das Sie die Apache-SparkSQL-Abfrage eingeben können. Sie können jedem Datensatz, der als Eingabe verwendet wird, Aliase zuweisen, um einfach die SQL-Abfrage zu unterstützen. Weitere Informationen über die SQL-Syntax finden Sie in der Spark-SQL-Dokumentation.

> **Note**
>
> Wenn Sie eine Spark-SQL-Transformation mit einer Datenquelle in einer VPC verwenden, fügen Sie eine AWS Glue VPC-Endpunkt zur VPC, die die Datenquelle enthält. Weitere Informationen zum Konfigurieren von Entwicklungsendpunkten finden Sie unter Hinzufügen eines Entwicklungsendpunkts, Einrichten Ihrer Umgebung für Entwicklungsendpunkte und Zugreifen auf den Entwicklungsendpunkt im AWS Glue -Entwicklerhandbuch.

Einen SQL-Transformationsknoten im Auftragsdiagramm verwenden

1. (Optional) Fügen Sie dem Auftragsdiagramm bei Bedarf einen Transformationsknoten hinzu. Wählen Sie SQL Query als Knotentyp aus.

> **Note**
>
> Wenn Sie eine Datenvorschausitzung und einen benutzerdefinierten SQL- oder benutzerdefinierten Codeknoten verwenden, führt die Datenvorschausitzung den SQL- oder Codeblock unverändert für den gesamten Datensatz aus.

2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist oder Sie mehrere Eingaben für die SQL-Transformation wünschen, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll. Fügen Sie nach Bedarf zusätzliche übergeordnete Knoten hinzu.
3. Wählen Sie die Registerkarte Transform (Transformation) im Bereich mit den Knotendetails aus.
4. Die Quell-Datensätze für die SQL-Abfrage sind durch die Namen kenntlich gemacht, die Sie im Feld Name bei den einzelnen Knoten angeben. Wenn Sie diese Namen nicht verwenden möchten oder wenn die Namen nicht für eine SQL-Abfrage geeignet sind, können Sie jedem Datensatz einzeln einen Namen zuordnen. Die Konsole stellt Standardaliase bereit, z. B. `MyDataSource`.

   Wenn beispielsweise ein übergeordneter Knoten für den SQL-Transformationsknoten `Rename Org PK field` heißt, könnten Sie den Namen `org_table` mit diesem Datensatz assoziieren. Dieser Alias kann dann anstelle des Knotennamens in der SQL-Abfrage verwendet werden.
5. Im Texteingabefeld unter der Überschrift Code block (Code-Block) fügen bzw. geben Sie die SQL-Abfrage ein. Das Textfeld zeigt SQL-Syntaxmarkierung an und macht Vorschläge zu Stichwörtern.
6. Wählen Sie bei ausgewähltem SQL-Transformationsknoten die Option Output schema (Ausgabeschema) und dann Edit (Bearbeiten) aus. Geben Sie die Spalten und Datentypen an, die die Ausgabefelder der SQL-Abfrage beschreiben.

   Geben Sie das Schema mithilfe der folgenden Aktionen im Abschnitt Output schema (Ausgabeschema) auf der Seite an:

   - Um eine Spalte umzubenennen, bewegen Sie den Cursor bei der Spalte in das Textfeld Key (Schlüssel) (auch als field (Field) oder property key (Eigenschaftsschlüssel) bezeichnet) und geben Sie den neuen Namen ein.
   - Um den Datentyp für eine Spalte zu ändern, wählen Sie den neuen Datentypen für die Spalte aus der Dropdown-Liste aus.

- Um dem Schema eine neue oberste Spalte hinzuzufügen, wählen Sie die Schaltfläche für Überlauf
  (⋯)
  und wählen Sie dann Add root key (Rootschlüssel hinzufügen) aus. Neue Spalten werden oben im Schema hinzugefügt.
- Um eine Spalte aus dem Schema zu entfernen, wählen Sie das Symbol zum Löschen
  (🗑)
  ganz rechts beim Schlüsselnamen aus.

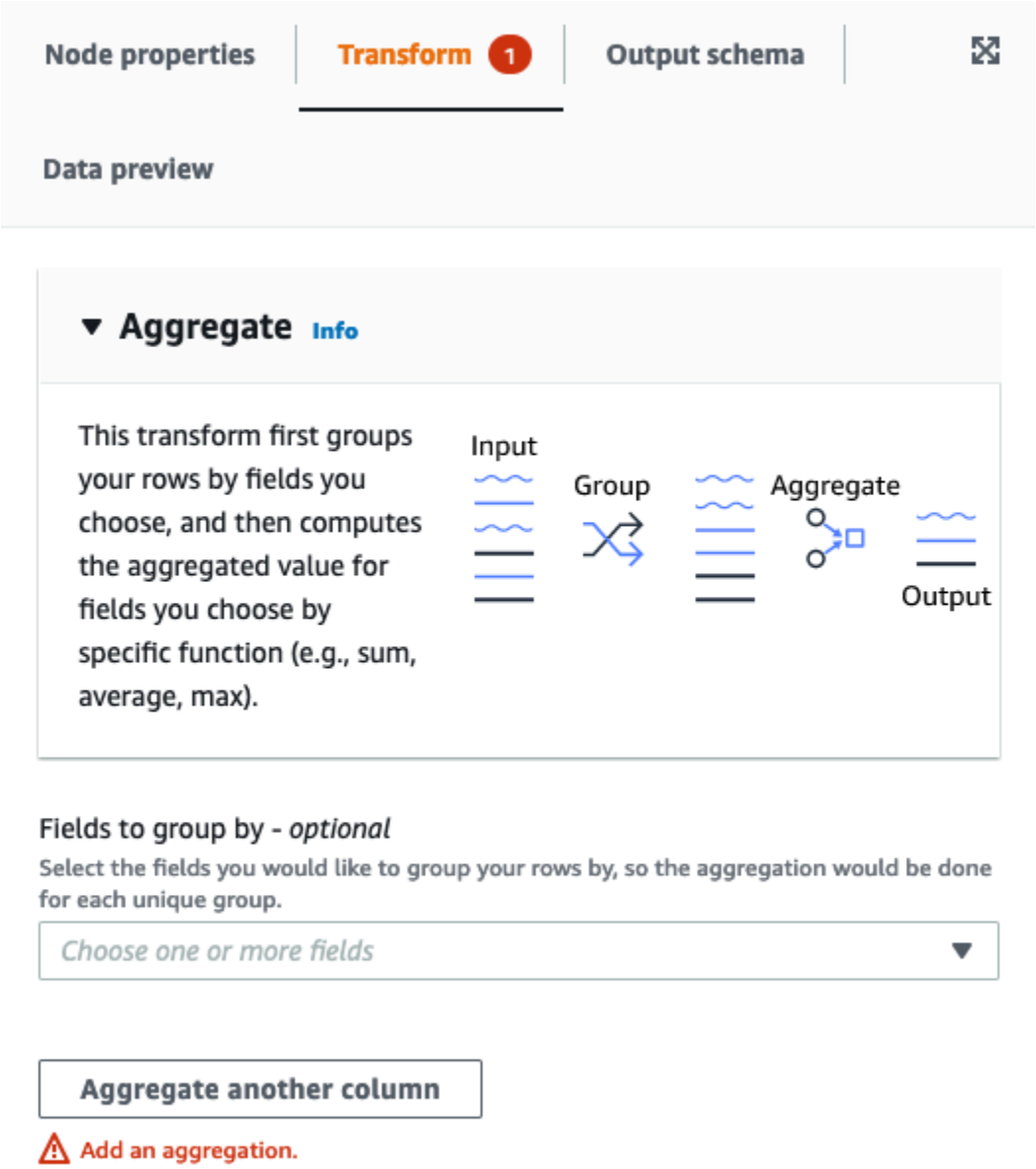
7. Wenn Sie die Angabe des Ausgabeschemas abgeschlossen haben, wählen Sie Apply (Anwenden) aus, um die Änderungen zu speichern und den Schema-Editor zu verlassen. Wenn Sie Änderungen nicht speichern möchten, wählen Sie Cancel (Abbrechen), um den Schema-Editor zu verlassen.
8. (Optional) Nachdem Sie die Knoteneigenschaften und Transformationseigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des geänderten Datensatzes. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Verwenden von Aggregate zum Durchführen zusammenfassender Berechnungen für ausgewählte Felder

Verwenden der Aggregate-Transformation

1. Fügen Sie den Aggregate-Knoten zum Auftragsdiagramm hinzu.
2. Wählen Sie in der Registerkarte Node properties (Knoteneigenschaften) die Felder aus, die gruppiert werden sollen, indem Sie das Dropdown-Feld auswählen (optional). Sie können mehrere Felder gleichzeitig auswählen oder nach einem Feldnamen suchen, indem Sie die Suchleiste verwenden.

   Wenn Felder ausgewählt sind, werden Name und Datentyp angezeigt. Wählen Sie im Feld „X“, um ein Feld zu entfernen.



3. Klicken Sie auf Aggregate another column (Weitere Spalte aggregieren). Es ist erforderlich, mindestens ein Feld auszuwählen.

4. Wählen Sie ein Feld im Dropdown-Menü Field to aggregate (Zu aggregierendes Feld).

5. Wählen Sie die Aggregationsfunktion aus, die auf das ausgewählte Feld angewendet werden soll:

   - avg - berechnet den Durchschnitt
   - countDistinct - berechnet die Anzahl der eindeutigen Nicht-Null-Werte
   - count - berechnet die Anzahl der Nicht-Null-Werte
   - first - gibt den ersten Wert zurück, der die Kriterien „Gruppieren nach“ erfüllt
   - last - gibt den letzten Wert zurück, der die Kriterien „Gruppieren nach“ erfüllt
   - kurtosis - berechnet die Schärfe des Peaks einer Frequenzverteilungskurve
   - max - gibt den höchsten Wert zurück, der die Kriterien „Gruppieren nach“ erfüllt
   - min - gibt den niedrigsten Wert zurück, der die Kriterien „Gruppieren nach“ erfüllt
   - Skewness - Maß für die Asymmetrie der Wahrscheinlichkeitsverteilung einer Normalverteilung
   - stddev_pop - berechnet die Standardabweichung der Bevölkerung und gibt die Quadratwurzel der Populationsvarianz zurück
   - sum - Die Summe aller Werte in der Gruppe
   - sumDistinct - Die Summe aller unterschiedlichen Werte in der Gruppe
   - var_samp - die Stichprobenvarianz der Gruppe (ignoriert Nullen)
   - var_pop - die Bevölkerungsvarianz der Gruppe (ignoriert Nullen)

Verschachtelte Strukturen verflachen

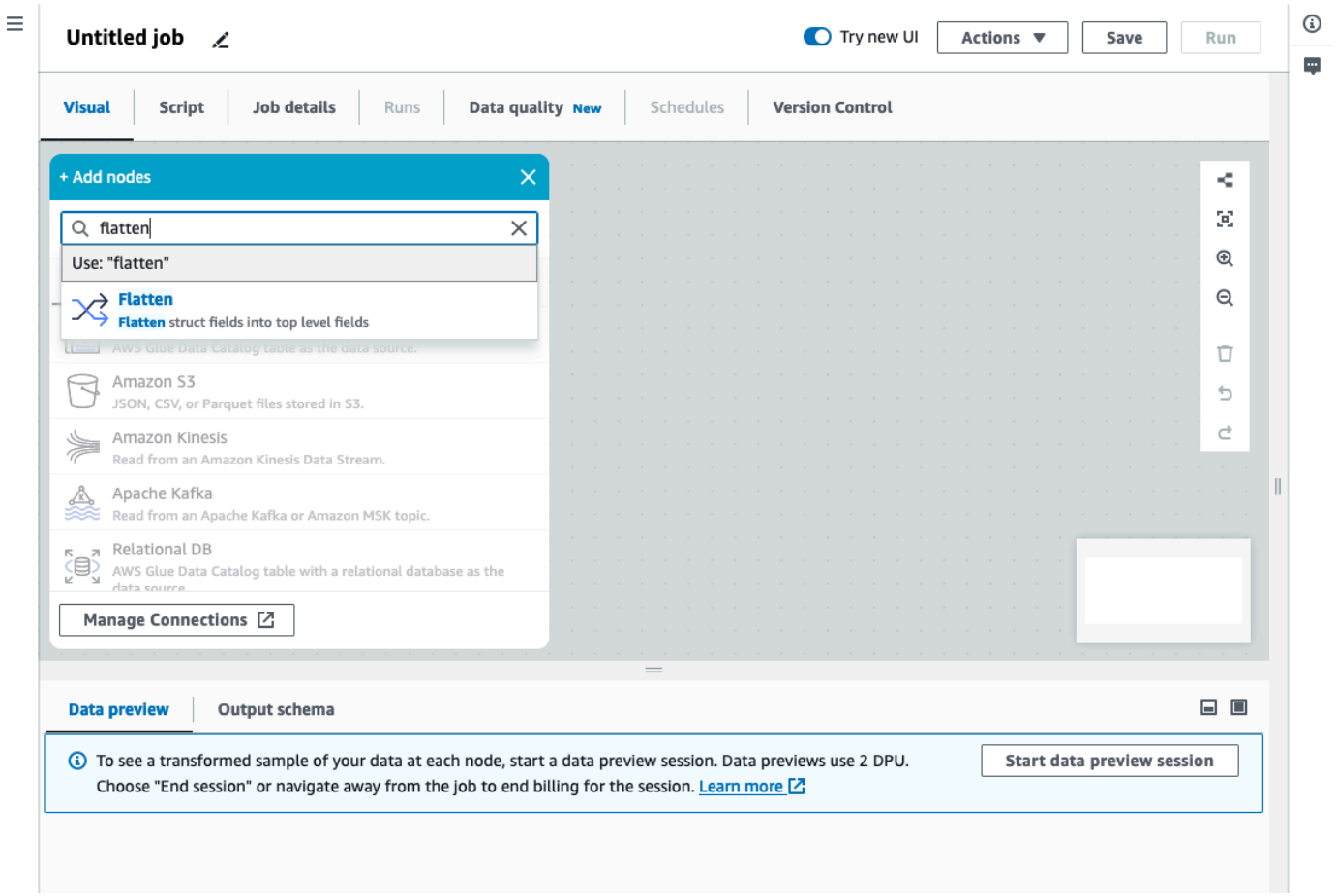
Verflacht die Felder verschachtelter Strukturen in den Daten, sodass sie zu Feldern der obersten Ebene werden. Die neuen Felder werden, um sie zu erreichen, anhand des Feldnamens benannt, dem die Namen der Strukturfelder vorangestellt sind, getrennt durch Punkte.

Zum Beispiel, wenn die Daten ein Feld vom Typ „Struct“ mit dem Namen „phone_numbers“ haben, das unter anderem eines vom Typ „Struct“ namens „home_phone“ mit zwei Feldern enthält: „Country_Code“ und „Number“. Nach dem Abflachen werden diese beiden Felder zu Feldern der obersten Ebene mit den Namen „phone_numbers.home_phone.country_code“ bzw. „phone_numbers.home_phone.number“.

Ihrem Auftragsdiagramm einen Verflachten Transformationsknoten hinzufügen

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie die Registerkarte Transformationen und dann Reduzieren aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen.

Sie können auch die Suchleiste verwenden, indem Sie „Reduzieren“ eingeben und dann auf den Knoten „Reduzieren“ klicken. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.



2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. (Optional) Auf der Registerkarte Transformieren können Sie die maximale Verschachtelungsebene einschränken, die verflacht werden soll. Wenn Sie diesen Wert beispielsweise auf 1 setzen, werden nur Strukturen der obersten Ebene verflacht. Wenn Sie das Maximum auf 2 setzen, werden die oberste Ebene und die direkt darunter liegenden Strukturen verflacht.

Hinzufügen einer UUID-Spalte

Wenn Sie eine Spalte UUID (Universally Unique Identified) hinzufügen, wird jeder Zeile eine eindeutige 36-stellige Zeichenfolge zugewiesen.

Ihrem Auftragsdiagramm einen UUID-Transformationsknoten hinzufügen

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie UUID aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. (Optional) Auf der Registerkarte Transformieren können Sie den Namen der neuen Spalte anpassen. Standardmäßig wird sie „uuid“ genannt.

Hinzufügen einer Identifikationsspalte

Weisen Sie jeder Zeile im Datensatz einen numerischen Identifikator zu.

Einen Identifikator-Transformationsknoten im Auftragsdiagramm hinzufügen

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Kennung aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. (Optional) Auf der Registerkarte Transformieren können Sie den Namen der neuen Spalte anpassen. Standardmäßig wird es „ID“ genannt.
4. (Optional) Wenn der Auftrag Daten inkrementell verarbeitet und speichert, möchten Sie vermeiden, dass dieselben IDs zwischen den Auftragsausführungen wiederverwendet werden.

   Markieren Sie auf der Registerkarte Transformieren die eindeutige Kontrollkästchen-Option. Es wird den Auftragszeitstempel in den Identifikator aufnehmen, sodass er zwischen mehreren

Durchläufen eindeutig ist. Um die größere Zahl zu berücksichtigen, wird die Spalte statt des Typs „long“ eine Dezimalzahl sein.

Eine Spalte in den Zeitstempeltyp konvertieren

Sie können die Transformation Zum Zeitstempel verwenden, um den Datentyp einer numerischen Spalte oder einer Zeichenkettenspalte in einen Zeitstempel zu ändern, sodass er mit diesem Datentyp gespeichert oder auf andere Transformationen angewendet werden kann, die einen Zeitstempel erfordert.

Dem Auftragsdiagramm einen Zum-Zeitstempel-Transformationsknoten hinzufügen

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Zum Zeitstempel aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformieren den Namen der zu konvertierenden Spalte ein.
4. Definieren Sie auf der Registerkarte Transformieren, wie die ausgewählte Spalte analysiert werden soll, indem Sie den Typ auswählen.

   Wenn der Wert eine Zahl ist, kann er in Sekunden (Unix/Python-Zeitstempel), Millisekunden oder Mikrosekunden ausgedrückt werden. Wählen Sie die entsprechende Option.

   Wenn der Wert eine formatierte Zeichenfolge ist, wählen Sie den Typ „iso“. Die Zeichenfolge muss einer der Varianten des ISO-Formats entsprechen, zum Beispiel: „2022-11-02T14:40:59.915Z“.

   Wenn Sie den Typ zu diesem Zeitpunkt noch nicht kennen oder verschiedene Zeilen unterschiedliche Typen verwenden, können Sie „automatische Erkennung“ wählen und das System wird seine beste Vermutung anstellen, mit einem kleinen Leistungsverlust.
5. (Optional) Auf der Registerkarte Transformieren können Sie, anstatt die ausgewählte Spalte zu konvertieren, eine neue erstellen und das Original beibehalten, indem Sie einen Namen für die neue Spalte eingeben.

Konvertieren Sie eine Zeitstempelspalte in eine formatierte Zeichenfolge

Formatieren Sie eine Zeitstempelspalte in eine Zeichenfolge, die auf einem Muster basiert. Sie können Zeitstempel formatieren verwenden, um Datum und Uhrzeit als Zeichenfolge mit dem gewünschten Format abzurufen. Sie können das Format mithilfe der Spark-Datumssyntax sowie der meisten Python-Datumscodes definieren.

Wenn Sie beispielsweise möchten, dass Ihre Datumszeichenfolge wie „2023-01-01 00:00“ formatiert wird, können Sie ein solches Format mit der Spark-Syntax als „yyyy-MM-dd HH:mm“ oder den entsprechenden Python-Datumscodes als „%Y-%m-%d %H: %M“ definieren

Ihrem Auftragsdiagramm einen Zeitstempel-formatieren-Transformationsknoten hinzufügen

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Zeitstempel formatieren aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformieren den Namen der zu konvertierenden Spalte ein.
4. Geben Sie auf der Registerkarte Transformieren das zu verwendende Zeitstempelformat-Muster ein, das mithilfe der Spark-Datumssyntax oder Python-Datumscodes ausgedrückt wird.
5. (Optional) Auf der Registerkarte Transformieren können Sie, anstatt die ausgewählte Spalte zu konvertieren, eine neue erstellen und das Original beibehalten, indem Sie einen Namen für die neue Spalte eingeben.

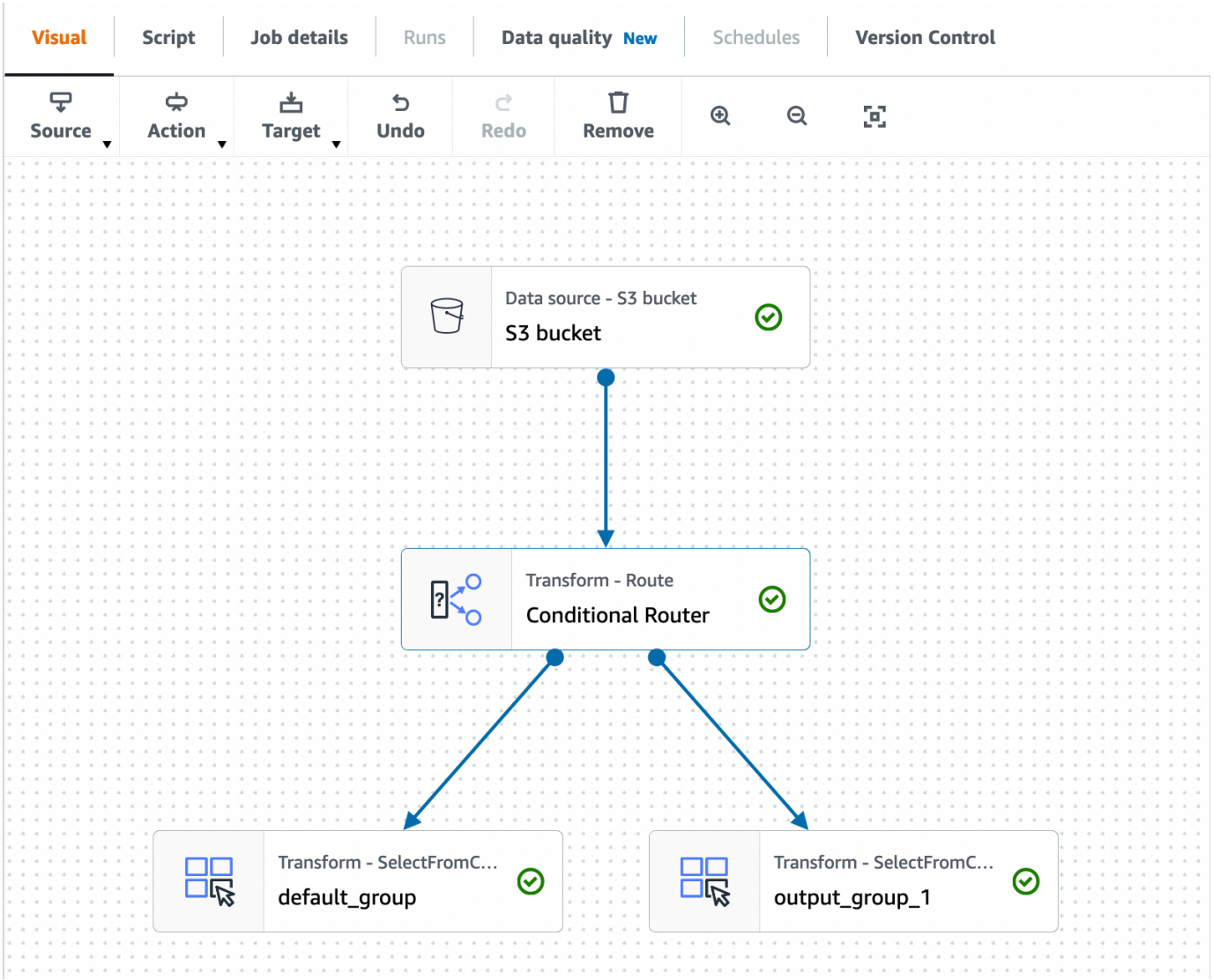
Erstellen einer bedingten Routertransformation

Mit der bedingten Router-Transformation können Sie mehrere Bedingungen auf eingehende Daten anwenden. Jede Zeile der eingehenden Daten wird anhand einer Gruppenfilterbedingung ausgewertet und zu der entsprechenden Gruppe verarbeitet. Wenn eine Zeile mehr als eine Gruppenfilterbedingung erfüllt, übergibt die Transformation die Zeile an mehrere Gruppen. Wenn eine Zeile keine Bedingung erfüllt, kann sie entweder gelöscht oder an eine Standardausgabegruppe weitergeleitet werden.

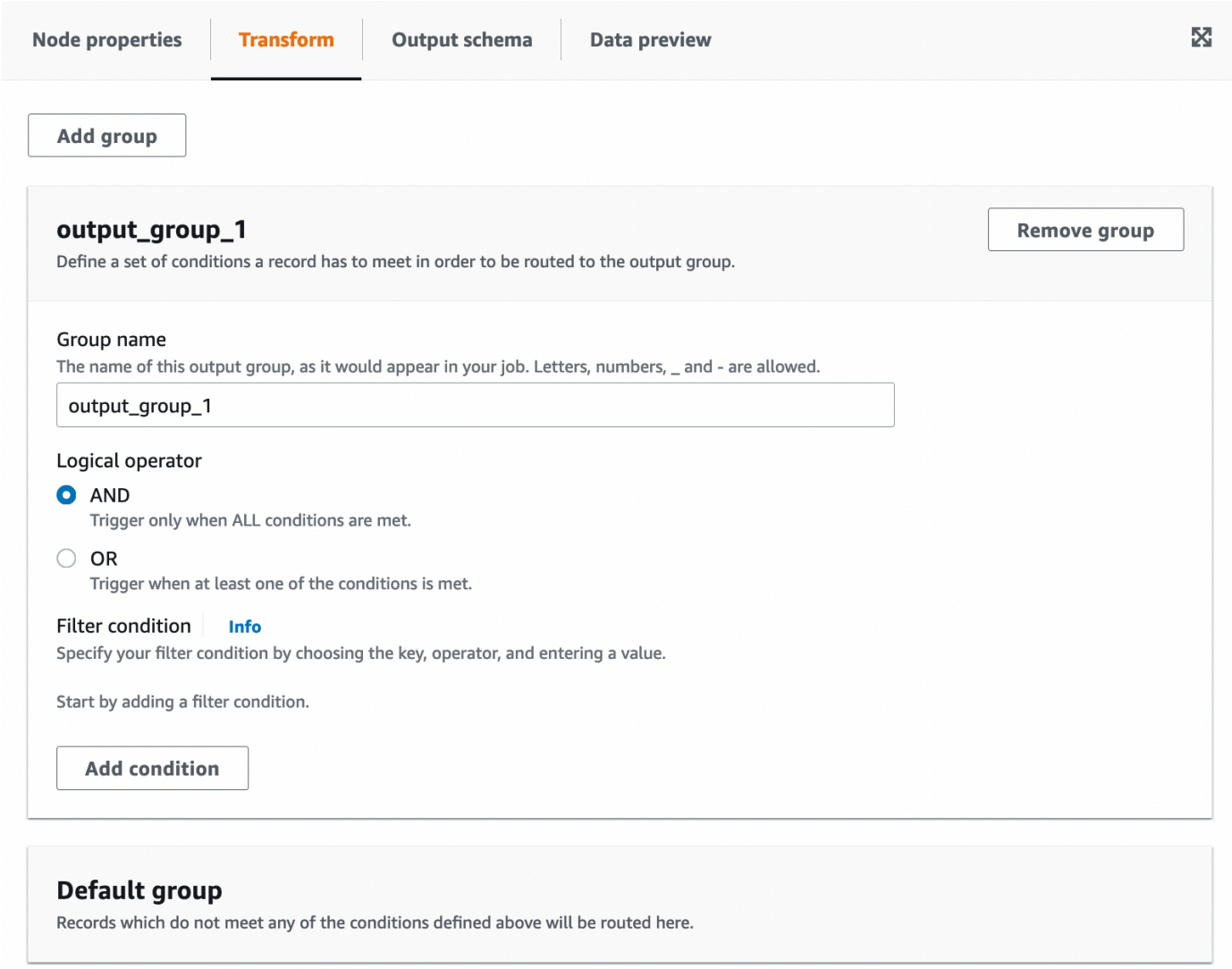
Diese Transformation ähnelt der Filtertransformation, ist jedoch nützlich für Benutzer, die dieselben Eingabedaten unter mehreren Bedingungen testen möchten.

Um eine bedingte Routertransformation hinzuzufügen:

1. Wählen Sie einen Knoten aus, auf dem Sie die bedingte Routertransformation durchführen möchten. Dies kann ein Quellknoten oder eine andere Transformation sein.
2. Wählen Sie Aktion und verwenden Sie dann die Suchleiste, um „Bedingter Router“ zu suchen und auszuwählen. Eine bedingte Router-Transformation wird zusammen mit zwei Ausgabeknoten hinzugefügt. Ein Ausgabeknoten, die „Standardgruppe“, enthält Datensätze, die keine der in den anderen Ausgangsknoten definierten Bedingungen erfüllen. Die Standardgruppe kann nicht bearbeitet werden.

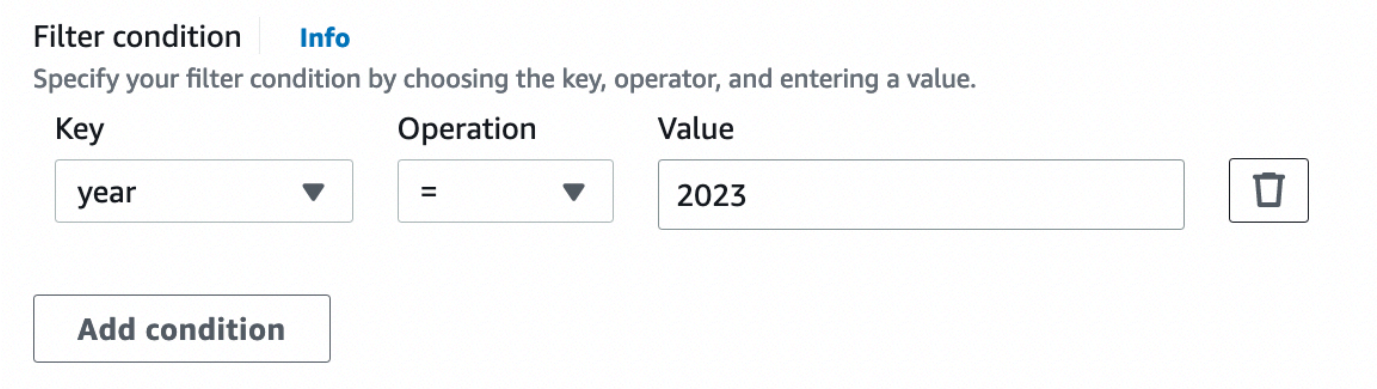


Sie können weitere Ausgabegruppen hinzufügen, indem Sie Gruppe hinzufügen wählen. Für jede Ausgabegruppe können Sie die Gruppe benennen und Filterbedingungen sowie einen logischen Operator hinzufügen.



3. Benennen Sie den Namen der Ausgabegruppe um, indem Sie einen neuen Namen für die Gruppe eingeben. AWS Glue Studio benennt Ihre Gruppen automatisch für Sie (zum Beispiel 'output_group_1').
4. Wählen Sie einen logischen Operator (AND, OR) und fügen Sie eine Filterbedingung hinzu, indem Sie den Schlüssel, den Vorgang und den Wert angeben. Logische Operatoren ermöglichen es Ihnen, mehr als eine Filterbedingung zu implementieren und den logischen Operator für jede von Ihnen angegebene Filterbedingung auszuführen.

   Bei der Angabe des Schlüssels können Sie aus den verfügbaren Schlüsseln in Ihrem Schema wählen. Sie können dann den verfügbaren Vorgang auswählen, der vom Typ des Schlüssels abhängt. Wenn der Schlüsseltyp beispielsweise „Zeichenfolge“ ist, steht als verfügbare Operation „Übereinstimmung“ zur Auswahl.



5. Geben Sie im Feld Wert den Wert ein. Wählen Sie Bedingung hinzufügen aus, um zusätzliche Bedingungen hinzuzufügen. Um einen Filter entfernen, klicken Sie auf das Papierkorbsymbol.

Verwenden der Transformation Spalten verketten zum Anfügen von Spalten

Mit der Verkettungs-Transformation können Sie eine neue Zeichenfolgenspalte erstellen, indem Sie die Werte anderer Spalten mit einem optionalen Abstandszeichen verwenden. Wenn wir beispielsweise eine verkettete Spalte „Datum“ als Verkettung von „Jahr“, „Monat“ und „Tag“ (in dieser Reihenfolge) mit „-“ als Abstandhalter definieren, erhalten wir:

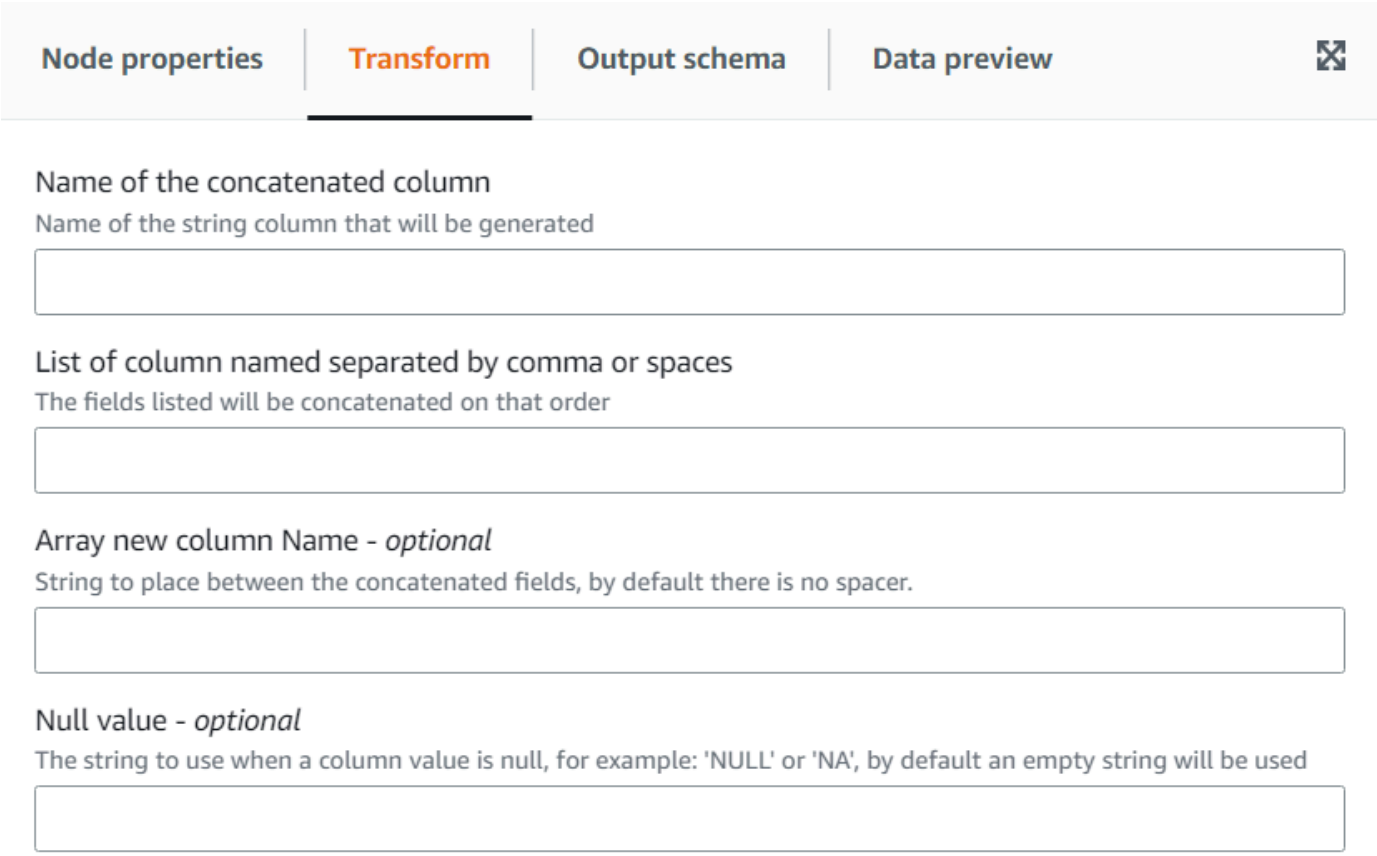
Tag	Monat	Jahr	date
01	01	2020	01.01.2020
02	01	2020	2020-01-02
03	01	2020	2020-01-03
04	01	2020	2020-01-04

So fügen Sie eine Verkettungs-Transformation hinzu:

1. Öffnen Sie das Bedienfeld Ressourcen. Wählen Sie dann Spalten verketten, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der

Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.

3. Geben Sie auf der Registerkarte Transformation den Namen der Spalte ein, die die verkettete Zeichenfolge sowie die zu verkettenden Spalten enthalten soll. Die Reihenfolge, in der Sie die Spalten im Dropdown-Menü überprüfen, entspricht der verwendeten Reihenfolge.



4. Abstandshalter – optional – Geben Sie eine Zeichenfolge ein, die zwischen den verketteten Feldern platziert werden soll. Standardmäßig ist kein Abstandshalter vorhanden.
5. Nullwert – optional – Geben Sie eine Zeichenfolge ein, die verwendet werden soll, wenn ein Spaltenwert null ist. Standardmäßig wird in den Fällen, in denen Spalten den Wert „NULL“ oder „NA“ haben, eine leere Zeichenfolge verwendet.

Verwenden der Transformation einer geteilten Zeichenfolge zum Aufteilen einer Zeichenfolgenspalte

Mit der Transformation einer geteilten Zeichenfolge können Sie eine Zeichenfolge in ein Array von Token aufteilen, indem Sie einen regulären Ausdruck verwenden, um zu definieren, wie die Aufteilung durchgeführt wird. Sie können die Spalte dann als Array-Typ beibehalten oder nach dieser

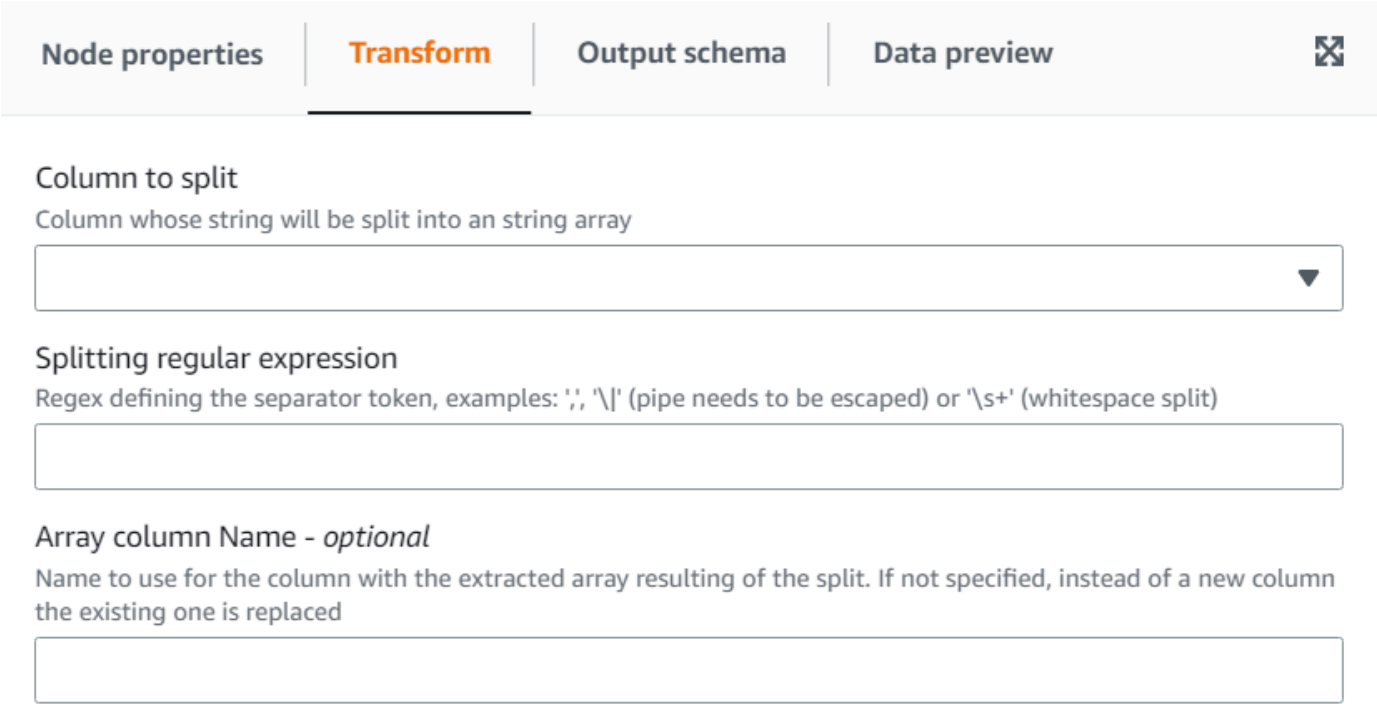
Transformation eine Array-zu-Spalten-Transformation anwenden, um die Array-Werte in Felder der obersten Ebene zu extrahieren, vorausgesetzt, dass jedes Token eine Bedeutung hat, die wir vorher kennen. Wenn die Reihenfolge der Token irrelevant ist (z. B. bei einer Reihe von Kategorien), können Sie außerdem die Explode-Transformation verwenden, um für jeden Wert eine separate Zeile zu generieren.

Sie können beispielsweise die Spalte „Kategorien“ mithilfe eines Kommas als Muster teilen, um eine Spalte „categories_arr“ hinzuzufügen.

product_id	categories	categories_arr
1	sports,winter	[sports, winter]
2	garden,tools	[garden, tools]
3	videogames	[videogames]
4	game,boardgame,social	[game, boardgame, social]

So fügen Sie eine Transformation einer geteilten Zeichenfolge hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Geteilte Zeichenfolge aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie auf der Registerkarte Transformieren die zu teilende Spalte aus und geben Sie das Muster ein, das zum Teilen der Zeichenfolge verwendet werden soll. In den meisten Fällen können Sie einfach die Zeichen eingeben, es sei denn, sie haben als regulärer Ausdruck eine besondere Bedeutung und müssen maskiert werden. Folgende Zeichen müssen maskiert werden: `\.[]{}()<>*+-=!?^$|` durch Einfügen eines umgekehrten Schrägstrichs vor dem Zeichen. Wenn Sie beispielsweise durch einen Punkt („.“) trennen möchten, müssen Sie `\.` eingeben. Ein Komma hat jedoch keine besondere Bedeutung und kann einfach so angegeben werden: `,`.



4. (Optional) Wenn Sie die ursprüngliche Zeichenfolgenspalte beibehalten möchten, können Sie einen Namen für eine neue Array-Spalte eingeben und so sowohl die ursprüngliche Zeichenfolgenspalte als auch die neue tokenisierte Array-Spalte beibehalten.

Verwenden Sie die Transformation Array zu Spalten, um die Elemente eines Arrays in Spalten der obersten Ebene zu extrahieren

Mit der Transformation Array zu Spalten können Sie einige oder alle Elemente einer Spalte vom Typ „Array“ in neue Spalten extrahieren. Die Transformation füllt die neuen Spalten so weit wie möglich, wenn das Array über genügend Werte zum Extrahieren verfügt. Optional übernimmt es die Elemente an den angegebenen Positionen.

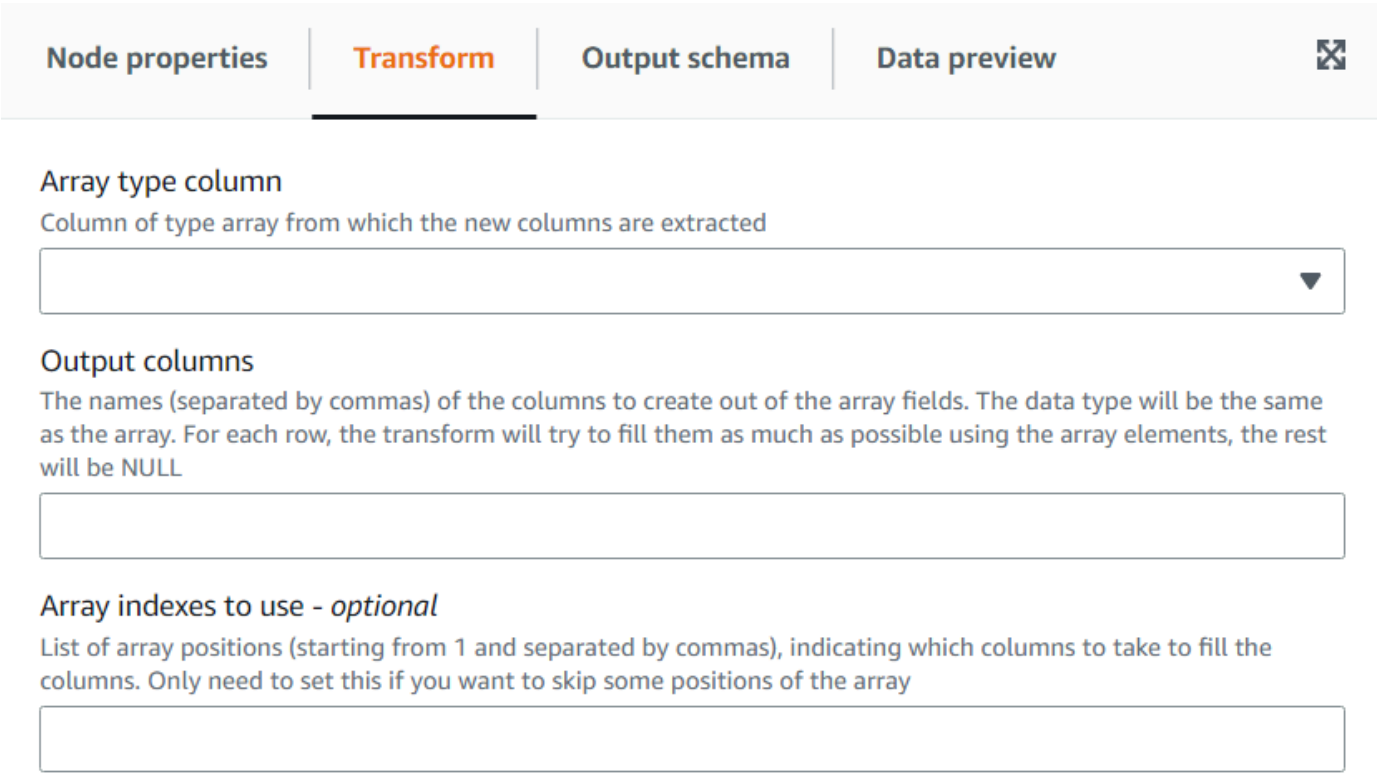
Wenn Sie beispielsweise über eine Array-Spalte „Subnetz“ verfügen, die das Ergebnis der Anwendung der Transformation einer geteilten Zeichenfolge auf ein IP-v4-Subnetz ist, können Sie die erste und vierte Position in die neuen Spalten „first_octect“ und „forth_octect“ extrahieren. Die Ausgabe der Transformation in diesem Beispiel wäre (beachten Sie, dass die letzten beiden Zeilen kürzere Arrays als erwartet haben):

Subnetz	first_octect	fourth_octect
[54, 240, 197, 238]	54	238

Subnetz	first_octect	fourth_octect
[192, 168, 0, 1]	192	1
[192, 168]	192	
[]		

So fügen Sie eine Transformation vom Typ Array zu Spalten hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Array zu Spalten aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie auf der Registerkarte Transformation die zu extrahierende Array-Spalte aus und geben Sie die Liste der neuen Spalten für die extrahierten Token ein.



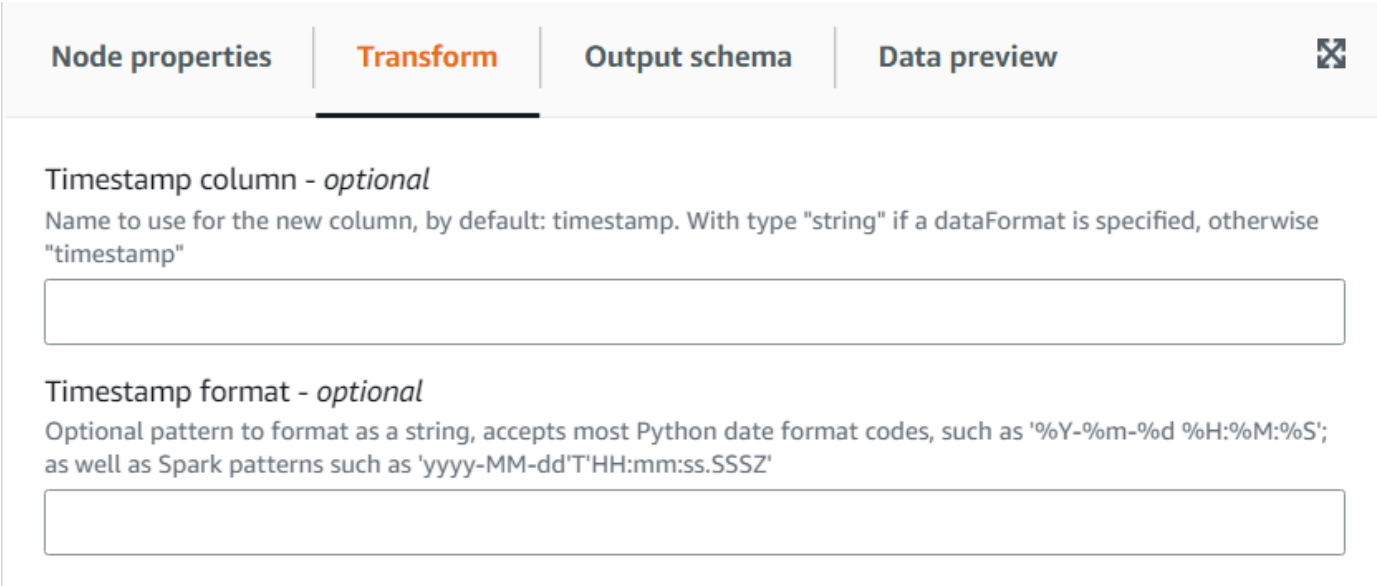
4. (Optional) Wenn Sie die Array-Token nicht für die Zuweisung zu Spalten verwenden möchten, können Sie die zu verwendenden Indizes angeben, die der Spaltenliste in derselben angegebenen Reihenfolge zugewiesen werden. Wenn die Ausgabespalten beispielsweise „column1, column2, column3“ und die Indizes „4, 1, 3“ sind, wird das vierte Element des Arrays in column1, das erste in columns2 und das dritte in column3 abgelegt (wenn das Array kürzer ist als die Indexnummer, wird ein NULL-Wert festgelegt).

Verwendung der Transformation zum Hinzufügen aktueller Zeitstempel

Mit der Transformation Hinzufügen aktueller Zeitstempel können Sie die Zeilen mit dem Zeitpunkt markieren, an dem die Daten verarbeitet wurden. Dies ist für Prüfzwecke oder zum Verfolgen der Latenz in der Datenpipeline nützlich. Sie können diese neue Spalte als Zeitstempel-Datentyp oder als formatierte Zeichenfolge hinzufügen.

So fügen Sie eine Transformation zum Hinzufügen aktueller Zeitstempel hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie dann Hinzufügen aktueller Zeitstempel aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.



3. (Optional) Geben Sie auf der Registerkarte Transformieren einen benutzerdefinierten Namen für die neue Spalte und ein Format ein, wenn die Spalte lieber eine formatierte Datumszeichenfolge sein soll.

Verwenden der Transformation „Zeilen zu Spalten pivotieren“

Mit der Transformation Zeilen zu Spalten pivotieren können Sie eine numerische Spalte aggregieren, indem Sie eindeutige Werte in ausgewählten Spalten rotieren, die zu neuen Spalten werden (wenn mehrere Spalten ausgewählt sind, werden die Werte verkettet, um die neuen Spalten zu benennen). Auf diese Weise werden Zeilen konsolidiert und gleichzeitig mehr Spalten mit teilweisen Aggregationen für jeden einzelnen Wert vorhanden. Wenn Sie beispielsweise über diesen Datensatz mit Verkäufen nach Monat und Land verfügen (zur einfacheren Veranschaulichung sortiert):

Jahr	Monat	country	Betrag
2020	. Jan.	uk	32
2020	. Jan.	de	42
2020	. Jan.	wir	64
2020	Februar	uk	67
2020	Februar	de	4
2020	Februar	de	7
2020	Februar	wir	6
2020	Februar	wir	12
2020	. Jan.	wir	90

Wenn Sie amount (Betrag) und country (Land) als Aggregationsspalten pivotieren, werden aus der ursprünglichen country-Spalte neue Spalten erstellt. In der folgenden Tabelle gibt es neue Spalten für de, uk und us anstelle der Spalte country (Land).

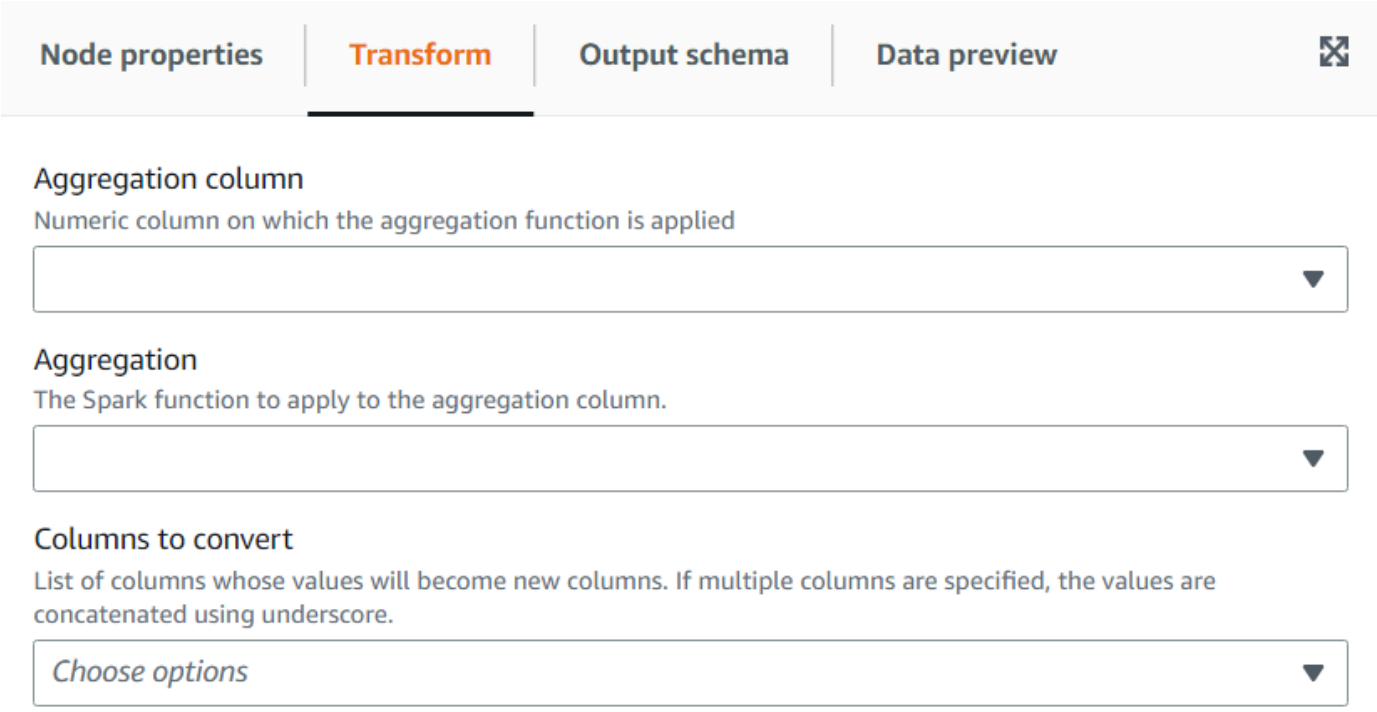
Jahr	Monat	de	uk	wir
2020	. Jan.	42	32	64
2020	. Jan.	11	67	18
2021	. Jan.			90

Wenn Sie stattdessen sowohl den Monat als auch den Landkreis pivotieren möchten, erhalten Sie eine Spalte für jede Kombination der Werte dieser Spalten:

Jahr	Jan_de	Jan_uk	Jan_us	Feb_de	Feb_uk	Feb_us
2020	42	32	64	11	67	18
2021			90			

So fügen Sie eine Transformation vom Typ „Zeilen zu Spalten pivotieren“ hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Zeilen zu Spalten pivotieren aus, um Ihrem Auftrag eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie auf der Registerkarte Transformation die numerische Spalte, die aggregiert werden soll, um die Werte für die neuen Spalten zu erzeugen. Wählen Sie die anzuwendende Aggregationsfunktion und die Spalte(n), deren eindeutige Werte in neue Spalten umgewandelt werden sollen.



Verwendung der Transformation „Spalten in Zeilen entpivotieren“

Mit der Transformation Entpivotieren können Sie Spalten in Werte neuer Spalten umwandeln und für jeden eindeutigen Wert eine Zeile generieren. Es ist das Gegenteil von Pivotieren, beachten Sie jedoch, dass es nicht gleichwertig ist, da es Zeilen mit identischen Werten, die aggregiert wurden, oder Aufteilungskombinationen nicht in die ursprünglichen Spalten trennen kann (das können Sie später mithilfe einer Aufteilungstransformation tun). Angenommen, Sie haben die folgende Tabelle:

Jahr	Monat	de	uk	wir
2020	. Jan.	42	32	64
2020	Februar	11	67	18
2021	. Jan.			90

Sie können die Spalten „de“, „uk“ und „us“ in eine Spalte „country“ mit dem Wert „amount“ entpivotieren und erhalten Folgendes (hier zur Veranschaulichung sortiert):

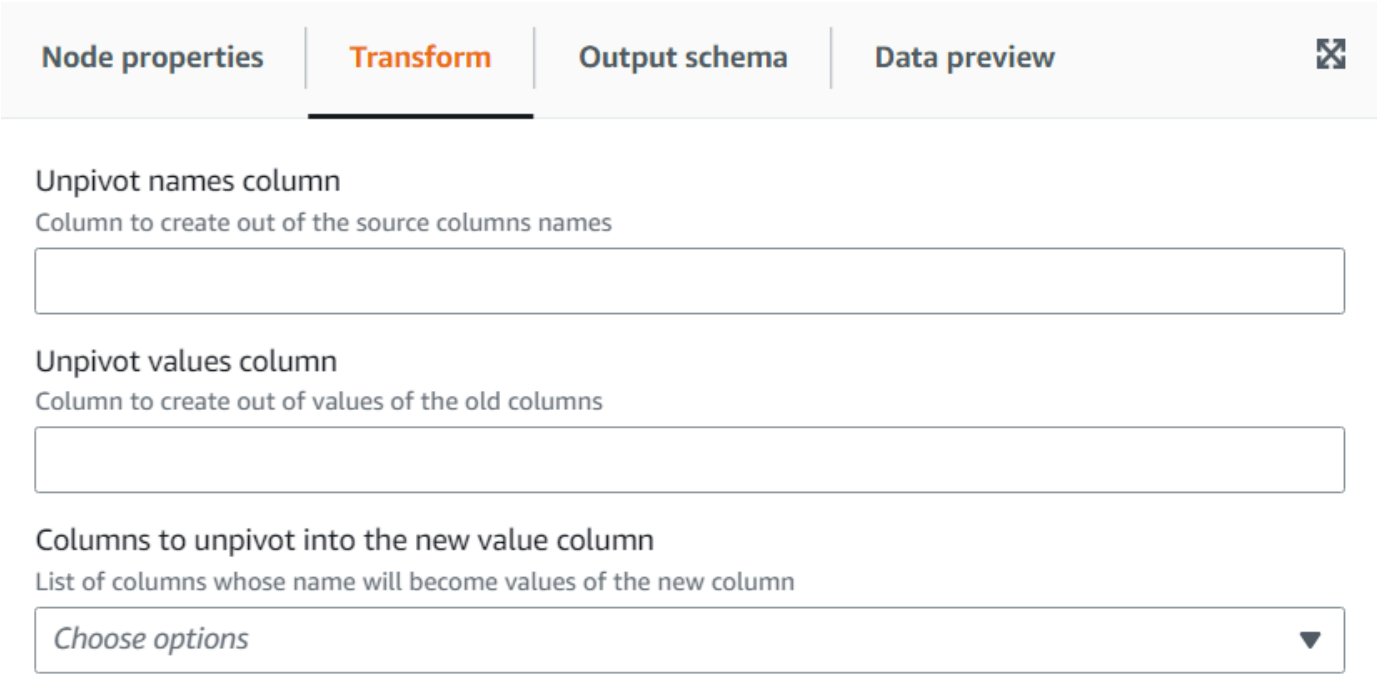
Jahr	Monat	country	Betrag
2020	. Jan.	uk	32
2020	. Jan.	de	42
2020	. Jan.	wir	64
2020	Februar	uk	67
2020	Februar	de	11
2020	Februar	wir	18
2021	. Jan.	wir	90

Beachten Sie, dass die Spalten mit einem NULL-Wert („de“ und „uk of Jan 2021“) standardmäßig nicht generiert werden. Sie können diese Option aktivieren, um Folgendes zu erhalten:

Jahr	Monat	country	Betrag
2020	. Jan.	uk	32
2020	. Jan.	de	42
2020	. Jan.	wir	64
2020	Februar	uk	67
2020	Februar	de	11
2020	Februar	wir	18
2021	. Jan.	wir	90
2021	. Jan.	de	
2021	. Jan.	uk	

So fügen Sie eine Transformation zum Entpivotieren von Spalten in Zeilen hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Spalten in Zeilen entpivotieren aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformieren die neuen Spalten ein, die erstellt werden sollen, um die Namen und Werte der zum Aufheben der Pivotierung ausgewählten Spalten aufzunehmen.

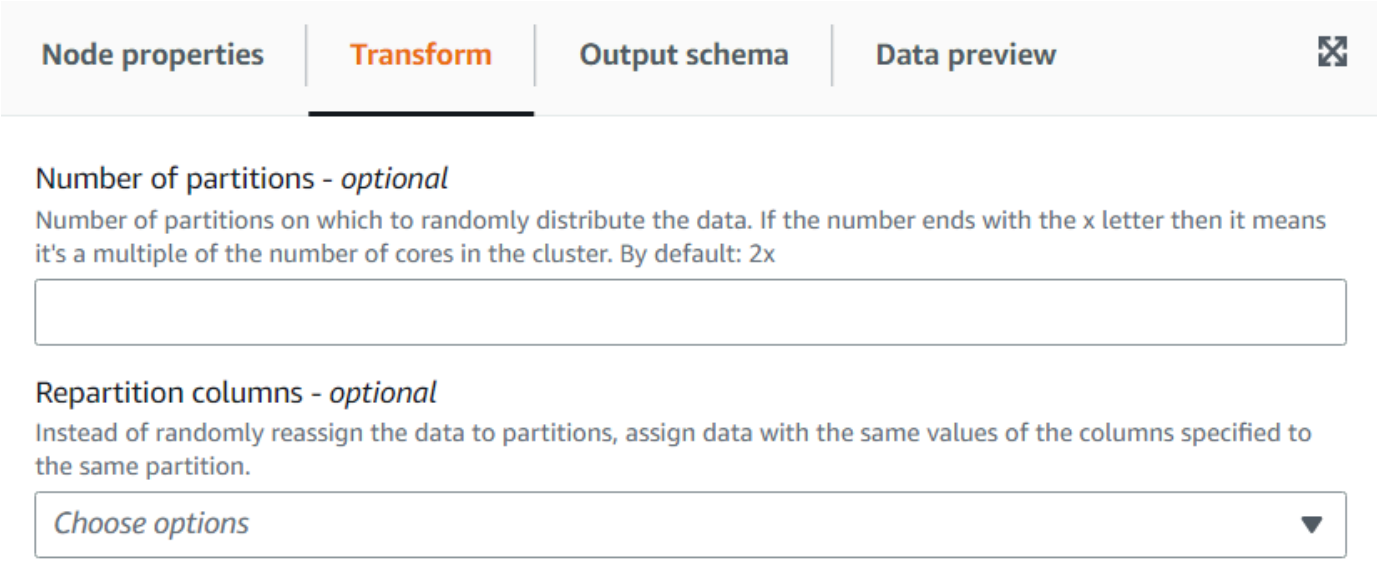


Nutzung der Transformation der Autobalance-Verarbeitung zur Optimierung Ihrer Laufzeit

Die Transformation Autobalance-Verarbeitung verteilt die Daten zwischen den Workern neu, um die Leistung zu verbessern. Dies hilft in Fällen, in denen die Daten unausgeglichen sind oder aufgrund ihrer Quelle keine ausreichende Parallelverarbeitung möglich ist. Dies ist häufig der Fall, wenn die Quelle GZIP-komprimiert oder JDBC ist. Die Neuverteilung von Daten verursacht geringe Leistungseinbußen, sodass die Optimierung diesen Aufwand möglicherweise nicht immer

kompensiert, wenn die Daten bereits gut abgeglichen sind. Darunter verwendet die Transformation die Neupartitionierung von Apache Spark, um Daten nach dem Zufallsprinzip einer Reihe von Partitionen zuzuordnen, die für die Cluster-Kapazität optimal sind. Für fortgeschrittene Benutzer besteht die Möglichkeit, mehrere Partitionen manuell einzugeben. Darüber hinaus kann es zur Optimierung des Schreibens partitionierter Tabellen verwendet werden, indem die Daten basierend auf angegebenen Spalten neu organisiert werden. Dies führt zu konsolidierteren Ausgabedateien.

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Autobalance-Verarbeitung aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. (Optional) Auf der Registerkarte Transformation können Sie eine Reihe von Partitionen eingeben. Im Allgemeinen wird empfohlen, diesen Wert dem System zu überlassen. Sie können jedoch den Multiplikator anpassen oder einen bestimmten Wert eingeben, wenn Sie dies steuern müssen. Wenn Sie die Daten nach Spalten partitioniert speichern möchten, können Sie dieselben Spalten als Neupartitionierungsspalten auswählen. Auf diese Weise wird die Anzahl der Dateien auf jeder Partition minimiert und vermieden, dass viele Dateien pro Partition vorhanden sind, was die Leistung der Tools, die diese Daten abfragen, beeinträchtigen würde.



Verwendung der Transformation für abgeleitete Spalten zum Kombinieren anderer Spalten

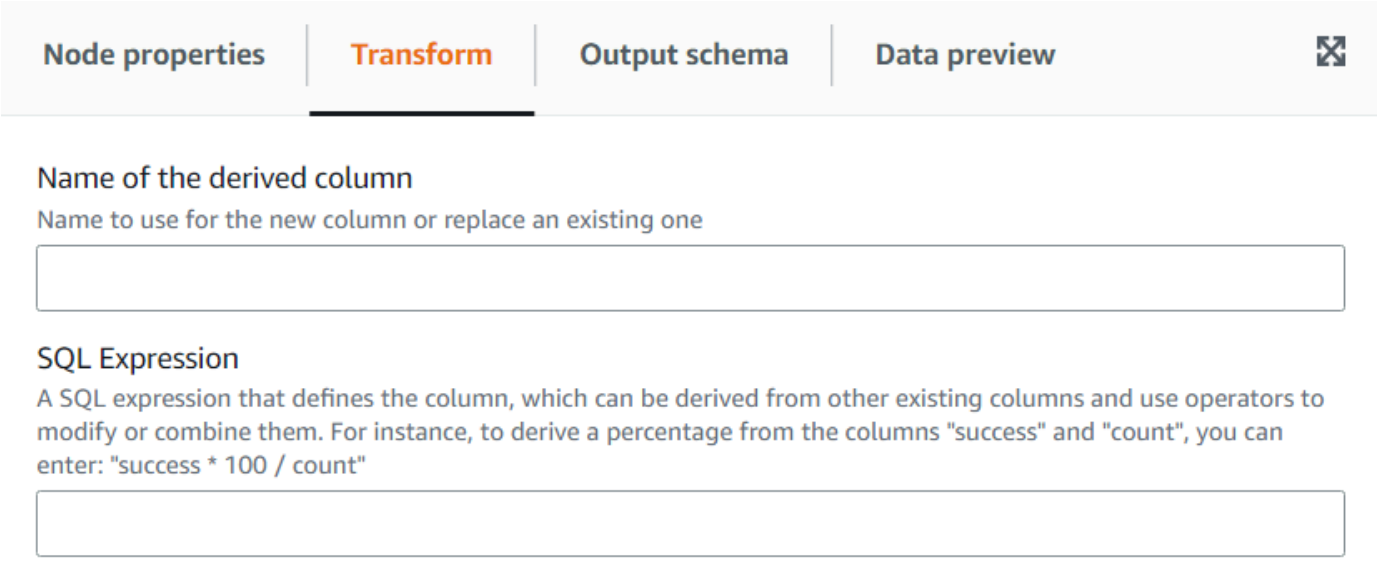
Mit der Transformation für Abgeleitete Spalten können Sie eine neue Spalte basierend auf einer mathematischen Formel oder einem SQL-Ausdruck definieren, in der Sie andere Spalten in den Daten sowie Konstanten und Literale verwenden können. Um beispielsweise eine „Prozent“-Spalte aus den Spalten „Erfolg“ und „Zähler“ abzuleiten, können Sie den SQL-Ausdruck eingeben: „Erfolg * 100 / Zähler || '%'“.

Beispielergebnis:

Erfolg	count	percentage
14	100	14%
6	20	3%
3	40	7,5%

So fügen Sie eine Transformation für abgeleitete Spalten hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Abgeleitete Spalte aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformation den Namen der Spalte und den Ausdruck für deren Inhalt ein.



Nutzung der Transformation Nachschlagen zum Hinzufügen übereinstimmender Daten aus einer Katalogtabelle

Mit der Transformation Nachschlagen können Sie Spalten aus einer definierten Katalogtabelle hinzufügen, wenn die Schlüssel mit den definierten Nachschlagsspalten in den Daten übereinstimmen. Dies entspricht der Durchführung einer Left-Outer-Zusammenführung zwischen den Daten und der Nachschlagetabelle unter Verwendung von Spalten als Bedingungsübereinstimmung.

So fügen Sie eine Transformation vom Typ Nachschlagen hinzu:

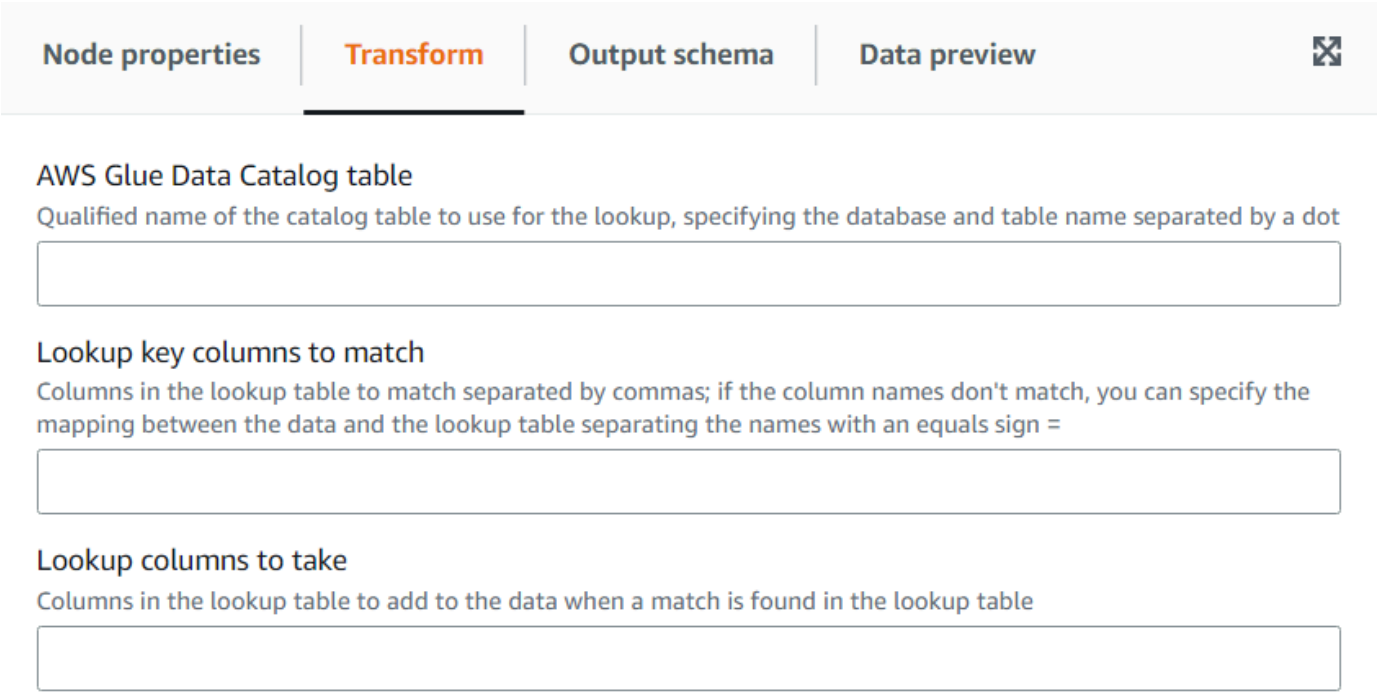
1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Nachschlagen aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformation den vollständig qualifizierten Katalogtabellennamen ein, der zum Durchführen der Suchvorgänge verwendet werden soll. Wenn Ihre Datenbank beispielsweise „mydb“ und Ihre Tabelle „mytable“ ist, geben Sie „mydb.mytable“ ein. Geben Sie dann die Kriterien ein, um in der Nachschlagetabelle eine Übereinstimmung zu finden, sofern der Nachschlageschlüssel zusammengesetzt ist. Geben Sie die Liste der Schlüsselspalten durch Kommas getrennt ein. Wenn eine

oder mehrere der Schlüsselspalten nicht den gleichen Namen haben, müssen Sie die Übereinstimmungszuordnung definieren.

Wenn die Datenspalten beispielsweise “user_id” und “region” lauten und in der Tabelle Benutzer die entsprechenden Spalten “id” und “region“ heißen, geben Sie in das Feld Abzugleichende Spalten ein: ”user_id=id, region“. Sie könnten region=region verwenden, dies ist jedoch nicht erforderlich, da beide identisch sind.

4. Geben Sie abschließend die Spalten ein, die aus der in der Nachschlagetabelle übereinstimmenden Zeile übernommen werden sollen, um sie in die Daten zu integrieren. Wenn keine Übereinstimmung gefunden wurde, werden diese Spalten auf NULL gesetzt.

> **Note**
>
> Unterhalb der Transformation Nachschlagen wird aus Effizienzgründen eine linke Zusammenführung verwendet. Wenn die Nachschlagetabelle über einen zusammengesetzten Schlüssel verfügt, stellen Sie sicher, dass die abzugleichenden Spalten so eingerichtet sind, dass sie mit allen Schlüsselspalten übereinstimmen, sodass nur eine Übereinstimmung erfolgen kann. Andernfalls werden mehrere Nachschlagezeilen übereinstimmen, was dazu führt, dass für jede dieser Übereinstimmungen zusätzliche Zeilen hinzugefügt werden.



Verwendung der Transformation „Array auflösen oder in Zeilen zuordnen“

Mit der Transformation Auflösen können Sie Werte aus einer verschachtelten Struktur in einzelne Zeilen extrahieren, die leichter zu bearbeiten sind. Im Fall eines Arrays generiert die Transformation eine Zeile für jeden Wert des Arrays und repliziert die Werte für die anderen Spalten in der Zeile. Im Falle einer Zuordnung generiert die Transformation für jeden Eintrag eine Zeile mit dem Schlüssel und dem Wert als Spalten sowie allen anderen Spalten in der Zeile.

Wenn wir beispielsweise diesen Datensatz haben, der eine „Kategorie“-Arrayspalte mit mehreren Werten enthält.

product_id	Kategorie
1	[sports, winter]
2	[garden, tools]
3	[videogames]
4	[game, boardgame, social]
5	[]

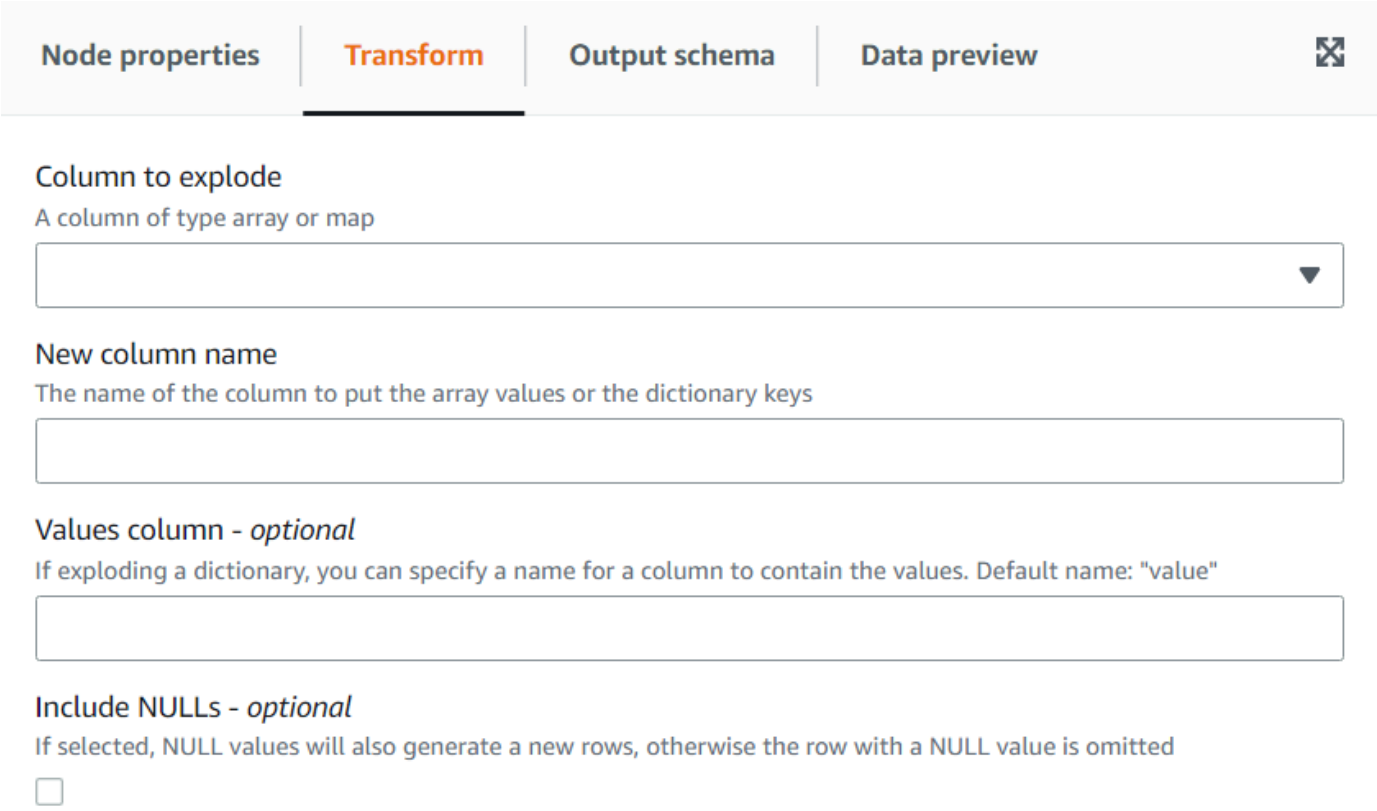
Wenn Sie die Spalte „Kategorie“ in eine Spalte mit demselben Namen auflösen, überschreiben Sie die Spalte. Sie können wählen, ob Sie Folgendes NULLs enthalten möchten, um Folgendes zu erhalten (zur Veranschaulichung bestellt):

product_id	Kategorie
1	sports
1	winter
2	garden
2	tool
3	videogames

product_id	Kategorie
4	game
4	boardgame
4	social
5	

So fügen Sie eine Transformation vom Typ „Array auflösen oder in Zeilen zuordnen“ hinzu:

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Array auflösen oder in Zeilen zuordnen aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Geben Sie auf der Registerkarte Knoteneigenschaften einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie auf der Registerkarte Transformation die Spalte aus, die Sie auflösen möchten (es muss sich um ein Array oder eine Zuordnung handeln). Geben Sie dann einen Namen für die Spalte für die Elemente des Arrays oder die Namen der Spalten für die Schlüssel und Werte ein, wenn Sie eine Zuordnung auflösen.
4. (Optional) Wenn auf der Registerkarte Transformation die aufzulösende Spalte NULL ist oder eine leere Struktur hat, wird sie im aufgelösten Datensatz standardmäßig weggelassen. Wenn Sie die Zeile beibehalten möchten (mit den neuen Spalten als NULL), aktivieren Sie „Einschließen NULLs“.

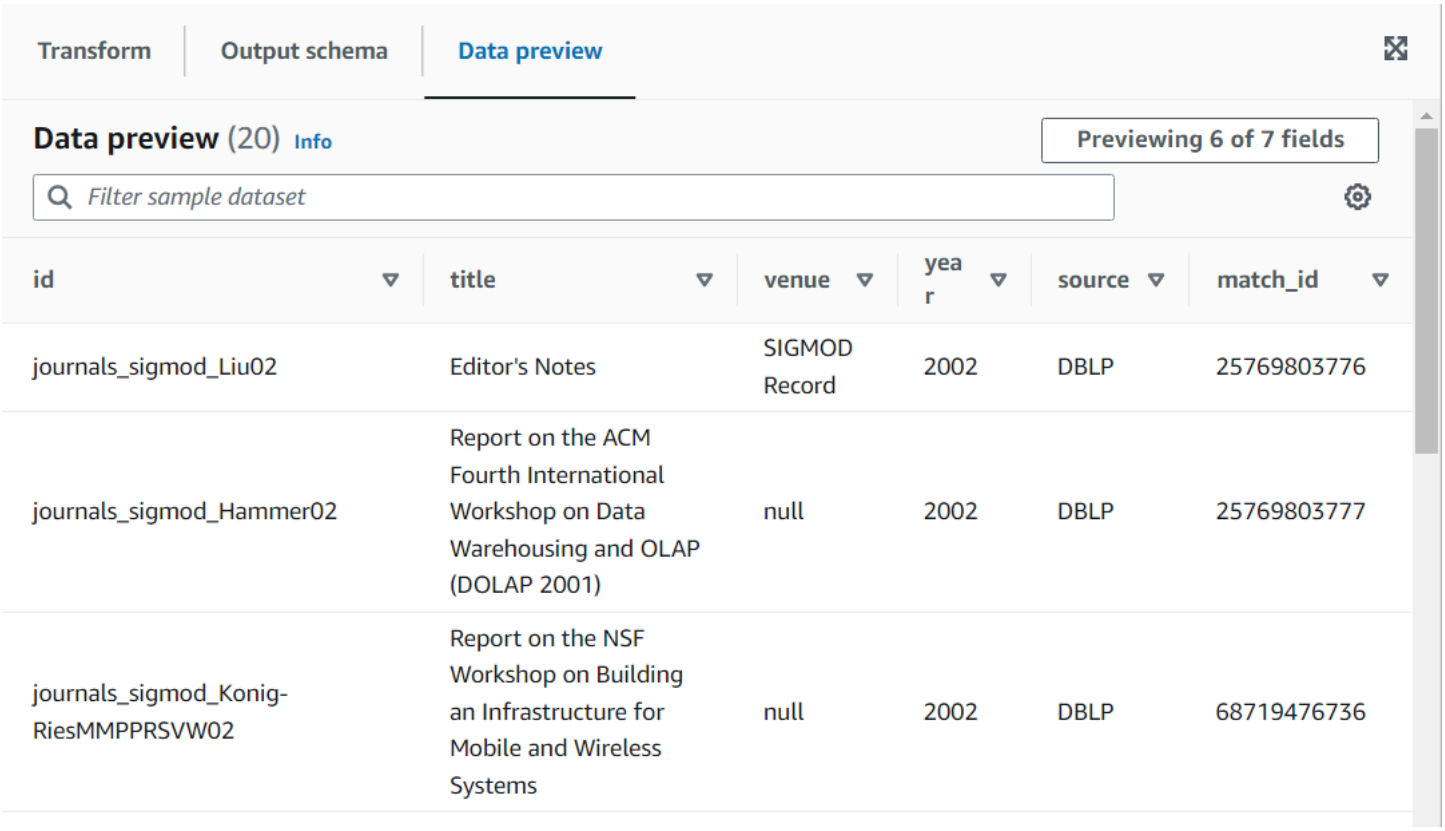


Verwendung der Transformation Datensatzabgleich zum Aufrufen einer vorhandenen Datenklassifizierungstransformation

Diese Transformation ruft eine vorhandene übereinstimmende Machine-Learning-Transformation zur Datenklassifizierung auf.

Die Transformation wertet die aktuellen Daten anhand des trainierten Modells anhand von Bezeichnungen aus. Eine Spalte „match_id“ wird hinzugefügt, um jede Zeile einer Gruppe von Elementen zuzuordnen, die basierend auf dem Algorithmustraining als gleichwertig gelten. Weitere Informationen finden Sie unter Datensatzabgleich mit Lake Formation FindMatches.

> **Note**
>
> Die vom Visual-Job AWS Glue verwendete Version von muss mit der Version übereinstimmen, die zur Erstellung der Transformation für den Datensatzabgleich AWS Glue verwendet wurde.



So fügen Sie Ihrem Auftragsdiagramm einen Transformationsknoten für den Datensatzabgleich hinzu

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Datensatzabgleich aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Im Bereich Knoteneigenschaften können Sie einen Namen für den Knoten im Auftragsdiagramm eingeben. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformation die ID ein, die Sie auf der Seite Machine-Learning-Transformationen finden:

4. (Optional) Auf der Registerkarte Transformation können Sie die Option zum Hinzufügen von Konfidenzwerten aktivieren. Auf Kosten zusätzlicher Rechenleistung schätzt das Modell eine Konfidenzbewertung für jede übereinstimmende Übereinstimmung in einer zusätzlichen Spalte.

Nullzeilen entfernen

Diese Transformation entfernt aus dem Datensatz Zeilen, deren Spalten alle Null sind. Darüber hinaus können Sie dieses Kriterium auf leere Felder erweitern, um Zeilen beizubehalten, in denen mindestens eine Spalte nicht leer ist.

So fügen Sie Ihrem Auftragsdiagramm einen Transformationsknoten Nullzeilen entfernen hinzu

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Nullzeilen entfernen aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Im Bereich Knoteneigenschaften können Sie einen Namen für den Knoten im Auftragsdiagramm eingeben. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. (Optional) Aktivieren Sie auf der Registerkarte Transformieren die Option Erweitert, wenn Sie möchten, dass Zeilen nicht nur nicht null, sondern auch nicht leer sind. Auf diese Weise werden leere Zeichenfolgen, Arrays oder Karten für diese Transformation als Nullen betrachtet.

Parsen einer Zeichenfolgenspalte mit JSON-Daten

Diese Transformation analysiert eine Zeichenfolgenspalte mit JSON-Daten und konvertiert sie in eine Struktur- oder Array-Spalte, je nachdem, ob es sich bei JSON um ein Objekt oder ein Array handelt. Optional können Sie sowohl die analysierte als auch die ursprüngliche Spalte beibehalten.

Das JSON-Schema kann mit optionaler Stichprobenauswahl bereitgestellt oder abgeleitet werden (im Fall von JSON-Objekten).

So fügen Sie Ihrem Auftragsdiagramm einen Knoten zum Parsen von JSON-Spalten hinzu

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie JSON-Spalte parsen aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.

2. Im Bereich Knoteneigenschaften können Sie einen Namen für den Knoten im Auftragsdiagramm eingeben. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Wählen Sie auf der Registerkarte Transformation die Spalte aus, die die JSON-Zeichenfolge enthält.
4. (Optional) Geben Sie auf der Registerkarte Transformation das Schema ein, dem die JSON-Daten folgen, und verwenden Sie dazu die SQL-Syntax, zum Beispiel: „field1 STRING, field2 INT“ im Fall eines Objekts oder „ARRAY<STRING>“ im Fall eines Arrays.

   Im Falle eines Arrays ist das Schema erforderlich, im Falle eines Objekts jedoch, wenn das Schema nicht angegeben ist, wird es anhand der Daten abgeleitet. Um die Auswirkungen der Ableitung des Schemas zu verringern (insbesondere bei einem großen Datensatz), können Sie das doppelte Lesen der gesamten Daten vermeiden, indem Sie ein Verhältnis der Stichproben eingeben, die zum Ableiten des Schemas verwendet werden sollen. Wenn der Wert kleiner als 1 ist, wird das entsprechende Verhältnis der Zufallsstichproben zur Ableitung des Schemas verwendet. Wenn die Daten zuverlässig sind und das Objekt zwischen den Zeilen konsistent ist, können Sie ein kleines Verhältnis wie 0,1 verwenden, um die Leistung zu verbessern.
5. (Optional) Auf der Registerkarte Transformation können Sie einen neuen Spaltennamen eingeben, wenn Sie sowohl die ursprüngliche Zeichenfolge-Spalte als auch die geparste Spalte beibehalten möchten.

Extrahieren eines JSON-Pfads

Diese Transformation extrahiert neue Spalten aus einer JSON-Zeichenfolgenspalte. Diese Transformation ist nützlich, wenn Sie nur wenige Datenelemente benötigen und nicht den gesamten JSON-Inhalt in das Tabellenschema importieren möchten.

So fügen Sie Ihrem Auftragsdiagramm einen Knoten für die Transformation vom Typ JSON-Pfad extrahieren hinzu

1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie JSON-Pfad extrahieren aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Im Bereich Knoteneigenschaften können Sie einen Namen für den Knoten im Auftragsdiagramm eingeben. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node

parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.

3. Wählen Sie auf der Registerkarte Transformation die Spalte aus, die die JSON-Zeichenfolge enthält. Geben Sie einen oder mehrere durch Kommas getrennte JSON-Pfadausdrücke ein, die jeweils darauf verweisen, wie ein Wert aus dem JSON-Array oder -Objekt extrahiert wird. Wenn die JSON-Spalte beispielsweise Objekte mit den Eigenschaften „prop_1“ und „prop2“ enthält, können Sie beide extrahieren und dabei ihre Namen „prop_1, prop_2“ angeben.

   Wenn das JSON-Feld Sonderzeichen enthält, können Sie beispielsweise zum Extrahieren der Eigenschaft aus diesem JSON `{"a. a": 1}` den Pfad `$['a. a']` verwenden. Die Ausnahme bildet das Komma, da es der Trennung von Pfaden vorbehalten ist. Geben Sie anschließend für jeden Pfad die entsprechenden Spaltennamen ein, getrennt durch Kommas.

4. (Optional) Auf der Registerkarte Transformation können Sie aktivieren, dass die JSON-Spalte nach der Extraktion gelöscht wird. Dies ist sinnvoll, wenn Sie den Rest der JSON-Daten nicht mehr benötigen, nachdem Sie die benötigten Teile extrahiert haben.

Extrahieren von Zeichenfolgenfragmenten mithilfe eines regulären Ausdrucks

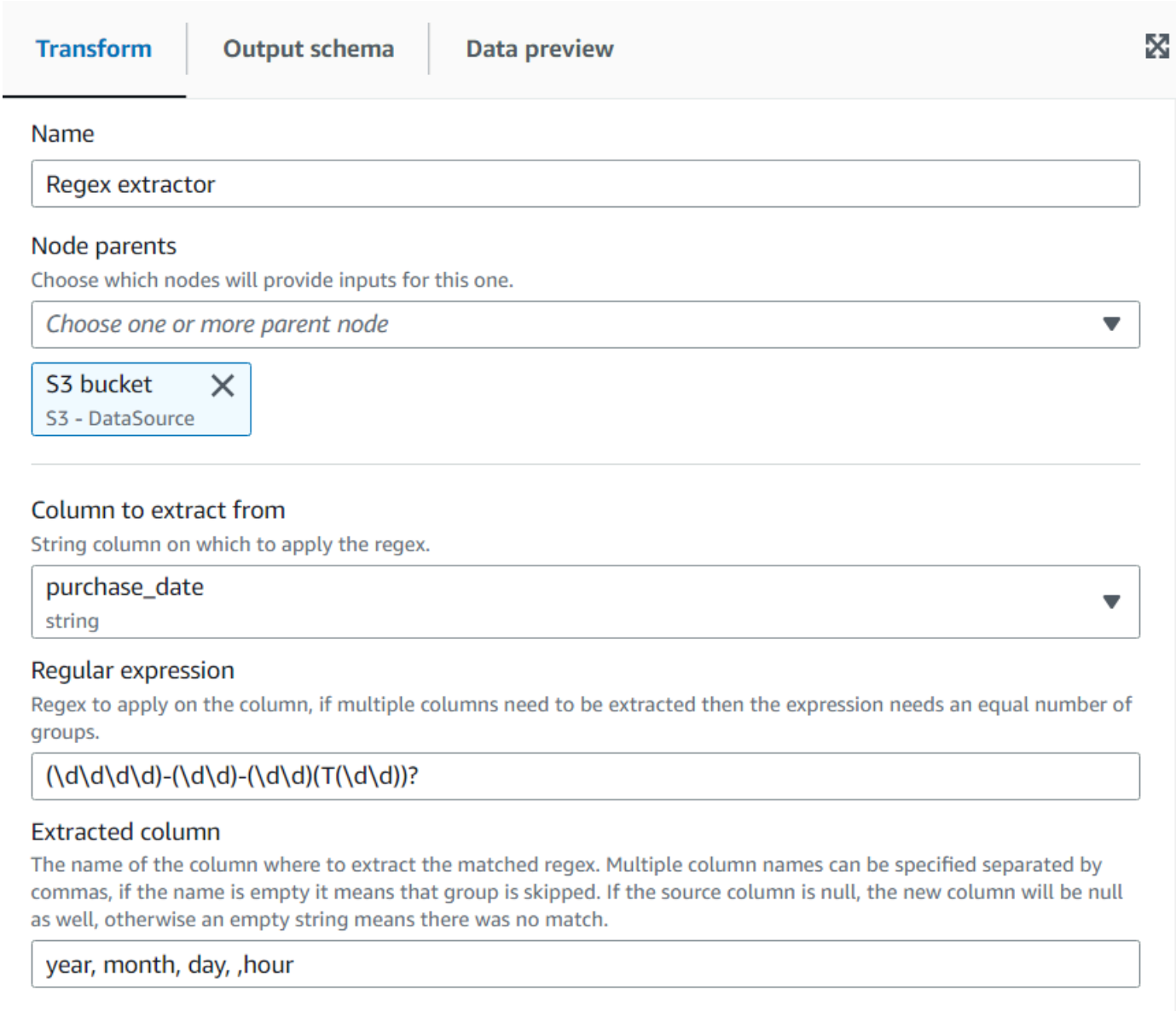
Diese Transformation extrahiert Zeichenfolgenfragmente mithilfe eines regulären Ausdrucks und erstellt daraus eine neue Spalte oder mehrere Spalten bei Verwendung von Regex-Gruppen.

So fügen Sie Ihrem Auftrag einen Knoten zum Transformieren des Regex-Extraktors hinzu

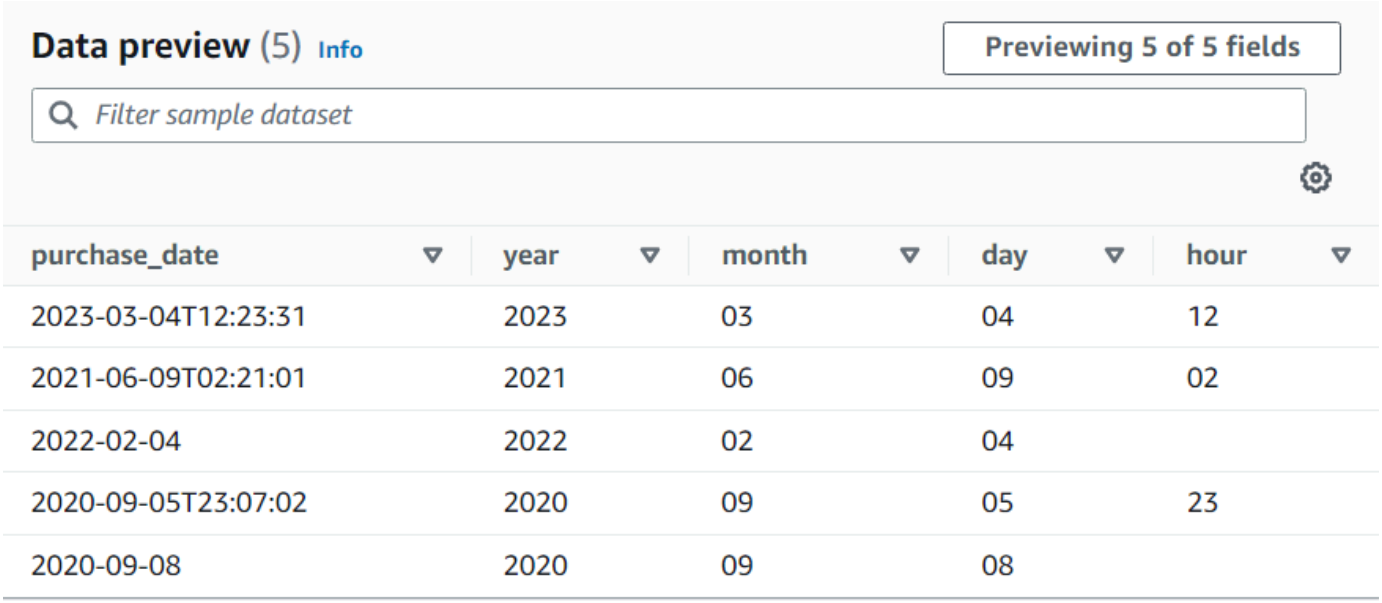
1. Öffnen Sie das Ressourcen-Bedienfeld und wählen Sie Regex-Extraktor aus, um Ihrem Auftragsdiagramm eine neue Transformation hinzuzufügen. Der Knoten, der zum Zeitpunkt des Hinzufügens ausgewählt wurde, ist sein übergeordneter Knoten.
2. Im Bereich Knoteneigenschaften können Sie einen Namen für den Knoten im Auftragsdiagramm eingeben. Falls noch kein übergeordneter Knoten ausgewählt ist, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.
3. Geben Sie auf der Registerkarte Transformieren den regulären Ausdruck und die Spalte ein, auf die er angewendet werden soll. Geben Sie dann den Namen der neuen Spalte ein, in der die passende Zeichenfolge gespeichert werden soll. Die neue Spalte ist nur dann null, wenn die Quellspalte null ist. Wenn der Regex nicht übereinstimmt, ist die Spalte leer.

Wenn der Regex Gruppen verwendet, muss ein entsprechender durch Komma getrennter Spaltenname vorhanden sein. Sie können Gruppen jedoch überspringen, indem Sie den Spaltennamen leer lassen.

Wenn Sie beispielsweise eine Spalte „purchase_date“ mit einer Zeichenfolge haben, die sowohl lange als auch kurze ISO-Datumsformate verwendet, dann möchten Sie Jahr, Monat, Tag und Stunde extrahieren, sofern verfügbar. Beachten Sie, dass die Stundengruppe optional ist. Andernfalls wären in den Zeilen, in denen sie nicht verfügbar ist, alle extrahierten Gruppen leere Zeichenfolgen (da der Regex nicht übereinstimmt). In diesem Fall möchten wir nicht, dass die Gruppe die Zeit optional macht, sondern die innere. Deshalb lassen wir den Namen leer und er wird nicht extrahiert (diese Gruppe würde das T-Zeichen enthalten).



Ergebnis der Datenvorschau:



purchase_date	year	month	day	hour
2023-03-04T12:23:31	2023	03	04	12
2021-06-09T02:21:01	2021	06	09	02
2022-02-04	2022	02	04	
2020-09-05T23:07:02	2020	09	05	23
2020-09-08	2020	09	08	

Erstellen einer benutzerdefinierten Transformation

Wenn Sie kompliziertere Transformationen für Ihre Daten durchführen müssen oder dem Datensatz Dateneigenschaftsschlüssel hinzufügen möchten, können Sie eine Transformation für benutzerdefinierten Code (Custom code) im Auftragsdiagramm hinzufügen. Mit dem Knoten „Custom code“ können Sie ein Skript eingeben, das die Transformation durchführt.

Wenn Sie benutzerdefinierten Code verwenden, müssen Sie die Änderungen, die durch den benutzerdefinierten Code an der Ausgabe vorgenommen werden, mit einem Schema-Editor angeben. Beim Bearbeiten des Schemas können Sie:

- Dateneigenschaftsschlüssel hinzufügen oder entfernen;
- den Datentyp von Dateneigenschaftsschlüsseln ändern;
- den Namen von Dateneigenschaftsschlüsseln ändern;
- einen verschachtelten Eigenschaftsschlüssel umstrukturieren.

Sie müssen eine SelectFromCollectionTransformation verwenden, um eine einzelne `DynamicFrame` aus dem Ergebnis Ihres benutzerdefinierten Transformationsknotens auszuwählen, bevor Sie die Ausgabe an einen Zielort senden können.

Mit den folgenden Aufgaben können Sie dem Auftragsdiagramm einen benutzerdefinierten Transformationsknoten hinzufügen.

Hinzufügen eines Transformationsknotens mit benutzerdefiniertem Code zum Auftragsdiagramm

Dem Auftragsdiagramm einen benutzerdefinierten Transformationsknoten hinzufügen

1. (Optional) Öffnen Sie das Bedienfeld Ressourcen und wählen Sie Benutzerdefinierte Transformation aus, um Ihrem Auftragsdiagramm eine benutzerdefinierte Transformation hinzuzufügen.
2. Geben Sie auf der Registerkarte Node properties (Knoteneigenschaften) einen Namen für den Knoten im Auftragsdiagramm ein. Falls noch kein übergeordneter Knoten ausgewählt ist oder Sie mehrere Eingaben für die benutzerdefinierte Transformation wünschen, wählen Sie in der Liste Node parents (Übergeordnete Knoten) einen Knoten aus, der als Eingabequelle für die Transformation verwendet werden soll.

Eingabe von Code für den benutzerdefinierten Transformationsknoten

Sie können Code in ein Eingabefeld eingeben oder einfügen. Der Auftrag verwendet diesen Code für die Datentransformation. Sie können einen Codeausschnitt entweder in Python oder Scala bereitstellen. Der Code sollte einen oder mehrere `DynamicFrames` als Eingabe nehmen und mehrere `DynamicFrames` zurückgeben.

Das Skript für einen benutzerdefinierten Transformationsknoten eingeben

1. Wenn der benutzerdefinierte Transformationsknoten im Auftragsdiagramm ausgewählt ist, wählen Sie die Registerkarte Transform (Transformation) aus.
2. Im Texteingabefeld unter der Überschrift Code block (Code-Block) fügen bzw. geben Sie den Code für die Transformation ein. Der Code, den Sie verwenden, muss zur Sprache passen, die auf der Registerkarte Job details (Auftragsdetails) für den Auftrag angegeben ist.

   Wenn Sie in Ihrem Code auf die Eingabeknoten verweisen, AWS Glue Studio benennt die vom Job `DynamicFrames` zurückgegebenen Diagrammknoten sequentiell in der Reihenfolge ihrer Erstellung. Verwenden Sie eines der folgenden Benennungsmethoden in Ihrem Code:

   - Klassische Codegenerierung — Verwenden Sie Funktionsnamen, um auf die Knoten in Ihrem Job-Diagramm zu verweisen.
     - Datenquellknoten: `DataSource0`, `DataSource1`, `DataSource2` usw.

- Transformationsknoten: `Transform0`, `Transform1`, `Transform2` usw.
- Neue Codegenerierung — Verwenden Sie den Namen, der auf der Registerkarte Node properties (Knoteneigenschaften) angegeben ist, angehängt mit '_node1','_node2', usw. Zum Beispiel: `S3bucket_node1`, `ApplyMapping_node2`, `S3bucket_node2`, `MyCustomNodeName_node1`.

Weitere Informationen zum neuen Codegenerator finden Sie unter Generierung des Skript-Code.

Die folgenden Beispiele zeigen das Format des Codes, der in das Codefeld eingegeben werden soll:

Python

Im folgenden Beispiel wird der erste erhaltene `DynamicFrame` zu einem `DataFrame` konvertiert, um die native Filtermethode anzuwenden (wobei nur Akten mit mehr als 1000 Stimmen beibehalten werden), und anschließend zurück zu einem `DynamicFrame` konvertiert und zurückgegeben.

```
def FilterHighVoteCounts (glueContext, dfc) -> DynamicFrameCollection:
    df = dfc.select(list(dfc.keys())[0]).toDF()
    df_filtered = df.filter(df["vote_count"] > 1000)
    dyf_filtered = DynamicFrame.fromDF(df_filtered, glueContext, "filter_votes")
    return(DynamicFrameCollection({"CustomTransform0": dyf_filtered}, glueContext))
```

Scala

Im folgenden Beispiel wird der erste erhaltene `DynamicFrame` zu einem `DataFrame` konvertiert, um die native Filtermethode anzuwenden (wobei nur Akten mit mehr als 1000 Stimmen beibehalten werden), und anschließend zurück zu einem `DynamicFrame` konvertiert und zurückgegeben.

```
object FilterHighVoteCounts {
  def execute(glueContext : GlueContext, input : Seq[DynamicFrame]) :
 Seq[DynamicFrame] = {
    val frame = input(0).toDF()
    val filtered = DynamicFrame(frame.filter(frame("vote_count") > 1000),
 glueContext)
    Seq(filtered)
  }
}
```

Bearbeiten des Schemas in einem benutzerdefinierten Transformationsknoten

Wenn Sie einen benutzerdefinierten Transformationsknoten verwenden, AWS Glue Studio kann die durch die Transformation erstellten Ausgabeschemas nicht automatisch ableiten. Sie verwenden den Schema-Editor, um die Schemaänderungen zu beschreiben, die vom benutzerdefinierten Transformationscode eingeführt werden.

Ein benutzerdefinierter Code-Knoten kann eine beliebige Anzahl von übergeordneten Knoten haben, die jeweils einen `DynamicFrame` als Eingabe für Ihren benutzerdefinierten Code bereitstellen. Ein benutzerdefinierter Code-Knoten gibt eine Sammlung von `DynamicFrames` zurück. Jeder `DynamicFrame`, der als Eingabe verwendet wird, besitzt ein zugehöriges Schema. Sie müssen ein Schema hinzufügen, das jeden einzelnen `DynamicFrame` beschreibt, der vom benutzerdefinierten Code-Knoten zurückgegeben wird.

> **Note**
>
> Wenn Sie Ihr eigenes Schema für eine benutzerdefinierte Transformation festlegen, AWS Glue Studio erbt keine Schemas von früheren Knoten. Um das Schema zu aktualisieren, wählen Sie den Knoten Benutzerdefinierte Transformation und anschließend die Registerkarte Datenvorschau aus. Sobald die Vorschau generiert wurde, wählen Sie „Vorschau-Schema verwenden“. Das Schema wird dann mithilfe der Vorschaudaten durch das Schema ersetzt.

Die Ausgabeschemata für einen benutzerdefinierten Transformationsknoten bearbeiten

1. Wenn der benutzerdefinierte Transformationsknoten im Auftragsdiagramm ausgewählt ist, wählen Sie im Bereich mit den Knotendetails die Registerkarte Output schema (Ausgabeschema) aus.
2. Wählen Sie Edit (Bearbeiten) aus, um Änderungen am Schema vorzunehmen.

   Wenn Sie verschachtelte Dateneigenschaftsschlüssel haben, z. B. ein Array oder ein Objekt, können Sie mit dem Symbol für Expand-Rows (Zeilen ausklappen) ( ) oben rechts in den Schemafenstern die Liste der untergeordneten Dateneigenschaftsschlüssel ausklappen. Nachdem Sie dieses Symbol ausgewählt haben, ändert es sich zu Collapse-Rows (Zeilen zuklappen)

( ), mit dem Sie die Liste der untergeordneten Eigenschaftsschlüssel zuklappen können.

3. Ändern Sie das Schema mithilfe der folgenden Aktionen im Abschnitt rechts auf der Seite:

   - Um einen Eigenschaftenschlüssel umzubenennen, bewegen Sie den Cursor in das Textfeld Key (Schlüssel) des jeweiligen Eigenschaftenschlüssels und geben Sie dann den neuen Namen ein.
   - Um den Datentyp eines Eigenschaftsschlüssels zu ändern, wählen Sie den neuen Datentyp für den Eigenschaftsschlüssel aus der Liste aus.
   - Um dem Schema einen neuen obersten Eigenschaftsschlüssel hinzuzufügen, wählen Sie das Symbol für Overflow (Überlauf) ( ) links von der Schaltfläche Cancel (Abbrechen) aus und dann Add root key (Rootschlüssel hinzufügen).
   - Um dem Schema einen untergeordneten Eigenschaftsschlüssel hinzuzufügen, wählen Sie das Symbol für Add-Key (Schlüssel hinzufügen) ( ) aus, das mit dem übergeordneten Schlüssel verknüpft ist. Geben Sie einen Namen für den untergeordneten Schlüssel ein und wählen Sie den Datentyp aus.
   - Um einen Eigenschaftsschlüssel aus dem Schema zu entfernen, wählen Sie das Symbol zum Löschen (Remove) ( ) ganz rechts neben dem Schlüsselnamen aus.

4. Wenn Ihr benutzerdefinierter Transformationscode mehrere `DynamicFrames` verwendet, können Sie zusätzliche Ausgabeschemata hinzufügen.

   - Um ein neues leeres Schema hinzuzufügen, wählen Sie das Symbol für Overflow (Überfluss) ( ) und anschließend Add output schema (Ausgabeschema hinzufügen) aus.
   - Um ein vorhandenes Schema in ein neues Ausgabeschema zu kopieren, stellen Sie sicher, dass das zu kopierende Schema in der Schemaauswahl angezeigt wird. Wählen Sie das Symbol für Overflow (Überfluss) ( ) und anschließend Duplicate (Duplizieren) aus.

Um ein Ausgabeschema zu löschen, stellen Sie sicher, dass das zu löschende Schema in der Schemaauswahl angezeigt wird. Wählen Sie das Symbol für Overflow (Überfluss) (⋯) und anschließend Delete (Löschen) aus.

5. Fügen Sie dem neuen Schema neue Root-Schlüssel hinzu oder bearbeiten Sie die duplizierten Schlüssel.
6. Wenn Sie die Ausgabeschemata modifizieren, wählen Sie die Schaltfläche Apply (Anwenden) aus, um die Änderungen zu speichern und den Schema-Editor zu verlassen.

   Wenn Sie Änderungen nicht speichern möchten, wählen Sie die Schaltfläche Cancel (Abbrechen) aus.

Konfigurieren der benutzerdefinierten Transformationsausgabe

Eine benutzerdefinierte Code-Transformation gibt eine Sammlung von `DynamicFrames` zurück, auch wenn es nur einen `DynamicFrame` in der Ergebnismenge gibt.

Die Ausgabe von einem benutzerdefinierten Transformationsknoten verarbeiten

1. Fügen Sie einen SelectFromCollectionTransformationsknoten hinzu, dessen übergeordneter Knoten der benutzerdefinierte Transformationsknoten ist. Aktualisieren Sie diese Transformation, um anzugeben, welchen Datensatz Sie verwenden möchten. Weitere Informationen finden Sie unter Wird verwendet SelectFromCollection , um auszuwählen, welcher Datensatz beibehalten werden soll.
2. Fügen Sie dem Auftragsdiagramm zusätzliche SelectFromCollectionTransformationen hinzu, wenn Sie zusätzliche Transformationen verwenden möchten, die vom benutzerdefinierten Transformationsknoten `DynamicFrames` erzeugt wurden.

   So könnten Sie etwa einen benutzerdefinierten Transformationsknoten hinzufügen, der einen Flug-Datensatz in mehrere Datensätze aufteilt aber einige der identifizierenden Eigenschaftsschlüssel in den einzelnen Ausgabeschemata dupliziert, z. B. Flugdatum oder Flugnummer. Sie fügen für jedes Ausgabeschema einen SelectFromCollectionTransformationsknoten hinzu, wobei der benutzerdefinierte Transformationsknoten der übergeordnete Knoten ist.

3. (Optional) Anschließend können Sie jeden SelectFromCollectionTransformationsknoten als Eingabe für andere Knoten im Job oder als übergeordnetes Element für einen Datenzielknoten verwenden.

Transformieren Sie Daten mit benutzerdefinierten visuellen Transformationen

Benutzerdefinierte visuelle Transformationen ermöglichen es Ihnen, Transformationen zu erstellen und sie für die Verwendung in verfügbar zu machen AWS Glue Studio Jobs. Mithilfe benutzerdefinierter visueller Transformationen können ETL-Entwickler, die möglicherweise nicht mit Programmieren vertraut sind, mithilfe der AWS Glue Studio -Schnittstelle implementieren.

Sie können eine benutzerdefinierte visuelle Transformation erstellen und sie dann auf Amazon S3 hochladen, um sie für die Verwendung im visuellen Editor in verfügbar zu machen AWS Glue Studio um mit diesen Jobs zu arbeiten.

Themen

- Erste Schritte mit benutzerdefinierten visuellen Transformationen
- Schritt 1. Erstellen einer JSON-Konfigurationsdatei
- Schritt 2. Implementieren der Transformationslogik
- Schritt 3. Validierung und Problembehebung benutzerdefinierter visueller Transformationen in AWS Glue Studio
- Schritt 4. Aktualisieren von benutzerdefinierten visuellen Transformationen nach Bedarf
- Schritt 5. Verwenden Sie benutzerdefinierte visuelle Transformationen in AWS Glue Studio
- Verwendungsbeispiele
- Beispiele für benutzerdefinierte visuelle Skripte
- Video

Erste Schritte mit benutzerdefinierten visuellen Transformationen

Um eine benutzerdefinierte visuelle Transformation zu erstellen, gehen Sie wie folgt vor.

- Schritt 1. Erstellen einer JSON-Konfigurationsdatei
- Schritt 2. Implementieren der Transformationslogik
- Schritt 3. Validieren der benutzerdefinierten visuellen Transformation

- Schritt 4. Aktualisieren der benutzerdefinierten visuellen Transformation nach Bedarf
- Schritt 5. Verwenden Sie die benutzerdefinierte visuelle Transformation in AWS Glue Studio

Richten Sie zunächst den Amazon-S3-Bucket ein und fahren Sie mit Step 1. (Schritt 1) fort. Create a JSON config file. (Erstellen Sie eine JSON-Konfigurationsdatei).

Voraussetzungen

Vom Kunden bereitgestellte Transformationen befinden sich in einem Kundenkonto. AWS Dieses Konto besitzt die Transformationen und verfügt daher über alle Berechtigungen, diese anzuzeigen (zu suchen und zu verwenden), zu bearbeiten oder zu löschen.

Um eine benutzerdefinierte Transformation zu verwenden AWS Glue Studio, müssen Sie in diesem AWS Konto zwei Dateien erstellen und in den Amazon S3 S3-Assets-Bucket hochladen:

- Python-Datei – enthält die Transformationsfunktion
- JSON-Datei – beschreibt die Transformation Dies wird auch als Konfigurationsdatei bezeichnet, die zum Definieren der Transformation erforderlich ist.

Um die Dateien miteinander zu koppeln, verwenden Sie für beide den gleichen Namen. Zum Beispiel:

- myTransform.JSON
- myTransform.py

Optional können Sie Ihrer benutzerdefinierten visuellen Transformation ein benutzerdefiniertes Symbol geben, indem Sie eine SVG-Datei bereitstellen, die das Symbol enthält. Um die Dateien miteinander zu koppeln, verwenden Sie denselben Namen für das Symbol:

- myTransform.svg

AWS Glue Studio ordnet sie automatisch anhand ihrer jeweiligen Dateinamen zu. Die Dateinamen dürfen für kein vorhandenes Modul identisch sein.

Empfohlene Konvention für den Namen der Transformationsdatei

AWS Glue Studio importiert Ihre Datei als Modul (zum Beispiel`import myTransform`) in Ihr Jobskript. Daher muss Ihr Dateiname denselben Benennungsregeln folgen, die für Python-Variablennamen (Bezeichner) festgelegt wurden. Sie müssen insbesondere entweder mit einem

Buchstaben oder einem Unterstrich beginnen und dann vollständig aus Buchstaben, Ziffern und/oder Unterstrichen bestehen.

> **Note**
>
> Stellen Sie sicher, dass Ihr Transformationsdateiname nicht im Widerspruch zu vorhandenen geladenen Python-Modulen (z. B. `sys, array, copy` usw.) steht, um unerwartete Laufzeitprobleme zu vermeiden.

Einrichten des Amazon-S3-Buckets

Von Ihnen erstellte Transformationen werden in Amazon S3 gespeichert und gehören Ihrem AWS Konto. Sie erstellen neue benutzerdefinierte visuelle Transformationen, indem Sie einfach Dateien (JSON und PY) in den Ordner für Amazon-S3-Komponenten hochladen, in dem derzeit alle Auftragsskripts gespeichert sind (z. B. `s3://aws-glue-assets-<accountid>-<region>/transforms`). Wenn Sie ein benutzerdefiniertes Symbol verwenden, laden Sie es ebenfalls hoch. Standardmäßig AWS Glue Studio liest alle .json-Dateien aus dem Ordner /transforms im selben S3-Bucket.

Schritt 1. Erstellen einer JSON-Konfigurationsdatei

Zum Definieren und Beschreiben Ihrer benutzerdefinierten visuellen Transformation ist eine JSON-Konfigurationsdatei erforderlich. Das Schema für die Konfigurationsdatei lautet wie folgt.

JSON-Dateistruktur

Felder

- `name: string` – (erforderlich) der Transformationssystemname, der zum Identifizieren von Transformationen verwendet wird. Befolgen Sie dieselben Benennungsregeln, die für Python-Variablennamen (Bezeichner) festgelegt wurden. Sie müssen insbesondere entweder mit einem Buchstaben oder einem Unterstrich beginnen und dann vollständig aus Buchstaben, Ziffern und/oder Unterstrichen bestehen.
- `displayName: string`— (optional) der Name der Transformation, der im AWS Glue Studio visueller Job-Editor. Wenn kein angegeben `displayName` ist, `name` wird der als Name der Transformation verwendet in AWS Glue Studio.
- `description: string`— (optional) Die Beschreibung der Transformation wird in angezeigt AWS Glue Studio und ist durchsuchbar.

- `functionName: string` – (erforderlich) Der Python-Funktionsname wird verwendet, um die aufzurufende Funktion im Python-Skript zu identifizieren.
- `path: string` – (optional) der vollständige Amazon-S3-Pfad zur Python-Quelldatei. Wenn nicht angegeben, AWS Glue verwendet den Dateinamenabgleich, um die Dateien „.json“ und „.py“ miteinander zu verknüpfen. Beispielsweise wird der Name der JSON-Datei, `myTransform.json`, mit der Python-Datei, `myTransform.py`, am selben Amazon-S3-Speicherort gekoppelt.
- `parameters: Array of TransformParameter object`— (optional) die Liste der Parameter, die angezeigt werden sollen, wenn Sie sie in der AWS Glue Studio visueller Editor.

TransformParameter Felder

- `name: string` – (erforderlich) der Parametername, der als benanntes Argument im Auftragsskript an die Python-Funktion übergeben wird. Befolgen Sie dieselben Benennungsregeln, die für Python-Variablennamen (Bezeichner) festgelegt wurden. Sie müssen insbesondere entweder mit einem Buchstaben oder einem Unterstrich beginnen und dann vollständig aus Buchstaben, Ziffern und/oder Unterstrichen bestehen.
- `displayName: string`— (optional) der Name der Transformation, der im AWS Glue Studio visueller Job-Editor. Wenn kein angegeben `displayName` ist, `name` wird der als Name der Transformation verwendet in AWS Glue Studio.
- `type: string` – (erforderlich) der Parametertyp, der gängige Python-Datentypen akzeptiert. Gültige Werte: 'str' | 'int' | 'float' | 'list' | 'bool'.
- `isOptional: boolean` – (optional) bestimmt, ob der Parameter optional ist. Standardmäßig sind alle Parameter erforderlich.
- `description: string`— (optional) Die Beschreibung wird in angezeigt AWS Glue Studio um dem Benutzer bei der Konfiguration des Transformationsparameters zu helfen.
- `validationType: string` – (optional) definiert, wie dieser Parameter validiert wird. Derzeit werden nur reguläre Ausdrücke unterstützt. Standardmäßig ist der Validierungstyp auf `RegularExpression` festgelegt.
- `validationRule: string` – (optional) regulärer Ausdruck, der verwendet wird, um die Formulareingabe vor dem Absenden zu validieren, wenn `validationType` auf `RegularExpression` festgelegt ist. Die Syntax regulärer Ausdrücke muss mit den RegExp Ecmascript-Spezifikationen kompatibel sein.
- `validationMessage: string` – (optional) die Meldung, die angezeigt werden soll, wenn die Validierung fehlschlägt.

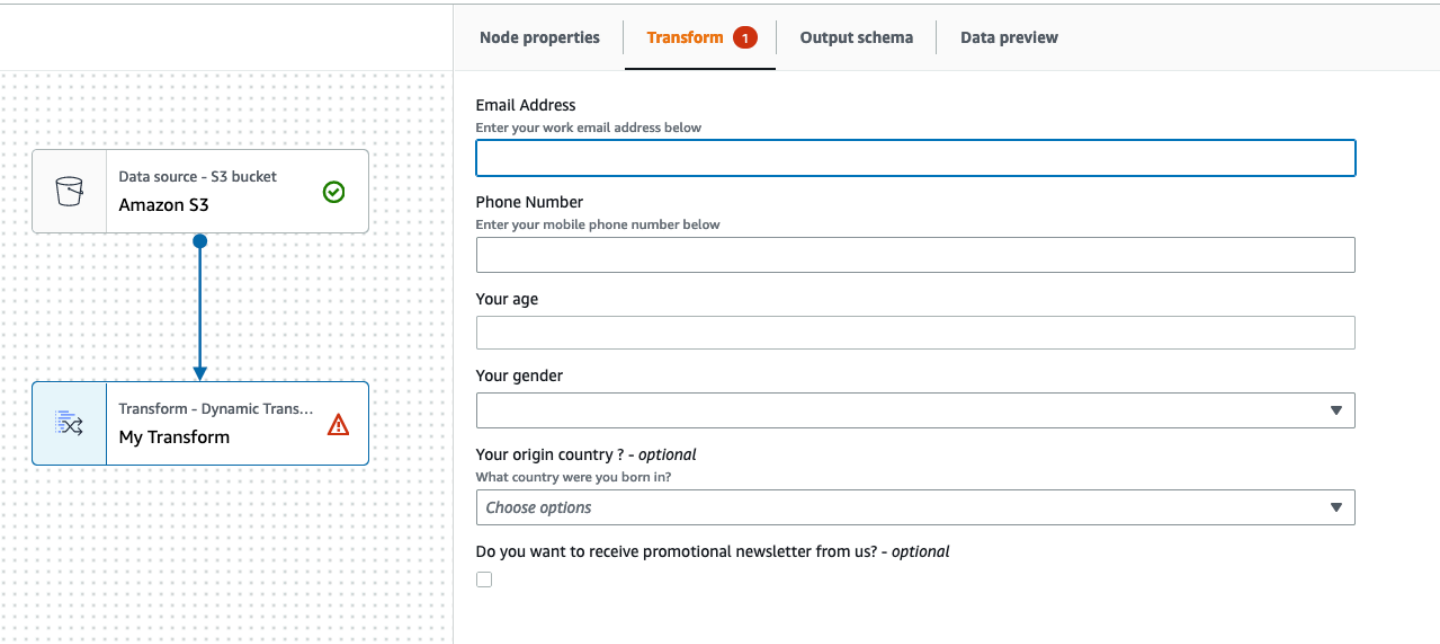
- `listOptions: An array of TransformParameterListOption object` ODER eine `string` oder der Zeichenfolgenwert „column“ – (optionale) Optionen zur Anzeige im Select- oder Multiselect-UI-Steuerelement. Diese akzeptieren eine Liste mit kommagetrennten Werten oder ein stark typisiertes JSON-Objekt vom Typ `TransformParameterListOption`. Es kann auch die Liste der Spalten aus dem übergeordneten Knotenschema dynamisch füllen, indem der Zeichenfolgenwert „column“ angegeben wird.
- `listType: string` – (optional) Definieren Sie Optionstypen für den Typ = 'list'. Gültige Werte: 'str' | 'int' | 'float' | 'list' | 'bool'. Parametertyp, der gängige Python-Datentypen akzeptiert.

TransformParameterListOption Felder

- `value: string | int | float | bool` – (erforderlich) Optionswert
- `label: string` – (optional) Optionsbezeichnung, die in der Auswahl-Dropdown-Liste angezeigt wird

Transformieren Sie Parameter in AWS Glue Studio

Standardmäßig sind Parameter erforderlich, es sei denn, sie sind in der JSON-Datei als `isOptional` gekennzeichnet. In AWS Glue Studio, Parameter werden auf der Registerkarte Transformieren angezeigt. Das Beispiel zeigt benutzerdefinierte Parameter wie E-Mail-Adresse, Telefonnummer, Ihr Alter, Ihr Geschlecht und Ihr Herkunftsland.

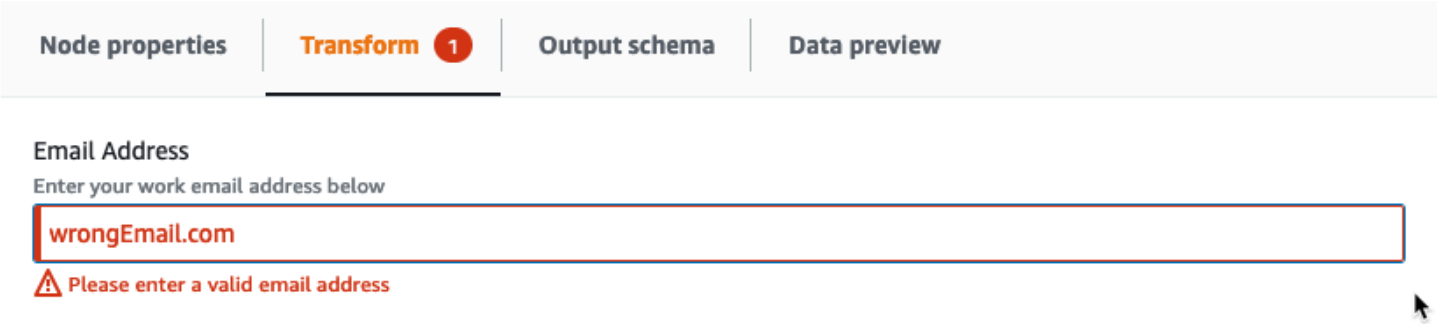


Sie können einige Validierungen erzwingen in AWS Glue Studio Verwenden Sie reguläre Ausdrücke in der JSON-Datei, indem Sie den `validationRule` Parameter und eine Bestätigungsnachricht in angeben. `validationMessage`

```
    "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
    "validationMessage": "Please enter a valid US number"
```

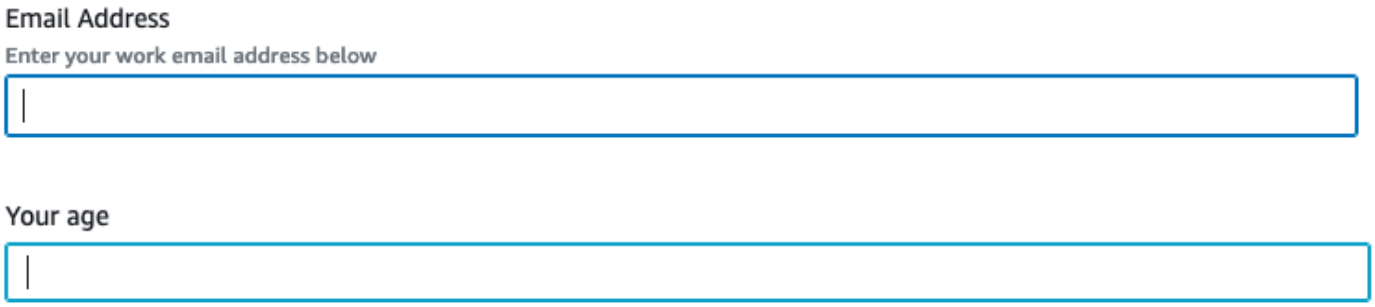
> **Note**
>
> Da die Validierung im Browser erfolgt, muss Ihre Syntax für reguläre Ausdrücke mit den RegExp Ecmascript-Spezifikationen kompatibel sein. Die Python-Syntax wird für diese regulären Ausdrücke nicht unterstützt.

Durch das Hinzufügen einer Validierung wird verhindert, dass der Benutzer den Job mit falschen Benutzereingaben speichert. AWS Glue Studio zeigt die Bestätigungsnachricht wie im Beispiel an:

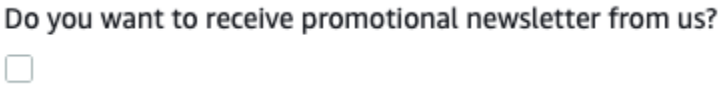


Parameter werden angezeigt in AWS Glue Studio basierend auf der Parameterkonfiguration.

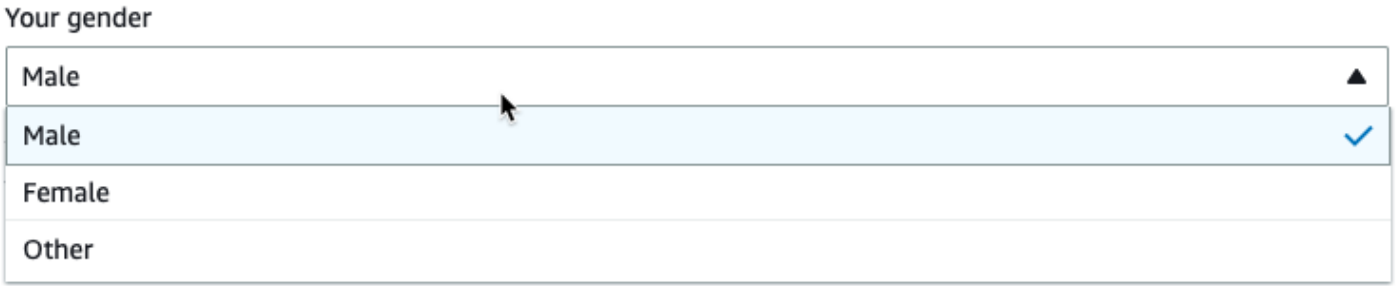
- Wenn `type` einer der folgenden Werte ist, z. B. `str`, `int` oder `float`, dann wird ein Texteingabefeld angezeigt. Der Screenshot zeigt beispielsweise Eingabefelder für die Parameter „E-Mail-Adresse“ und „Ihr Alter“.



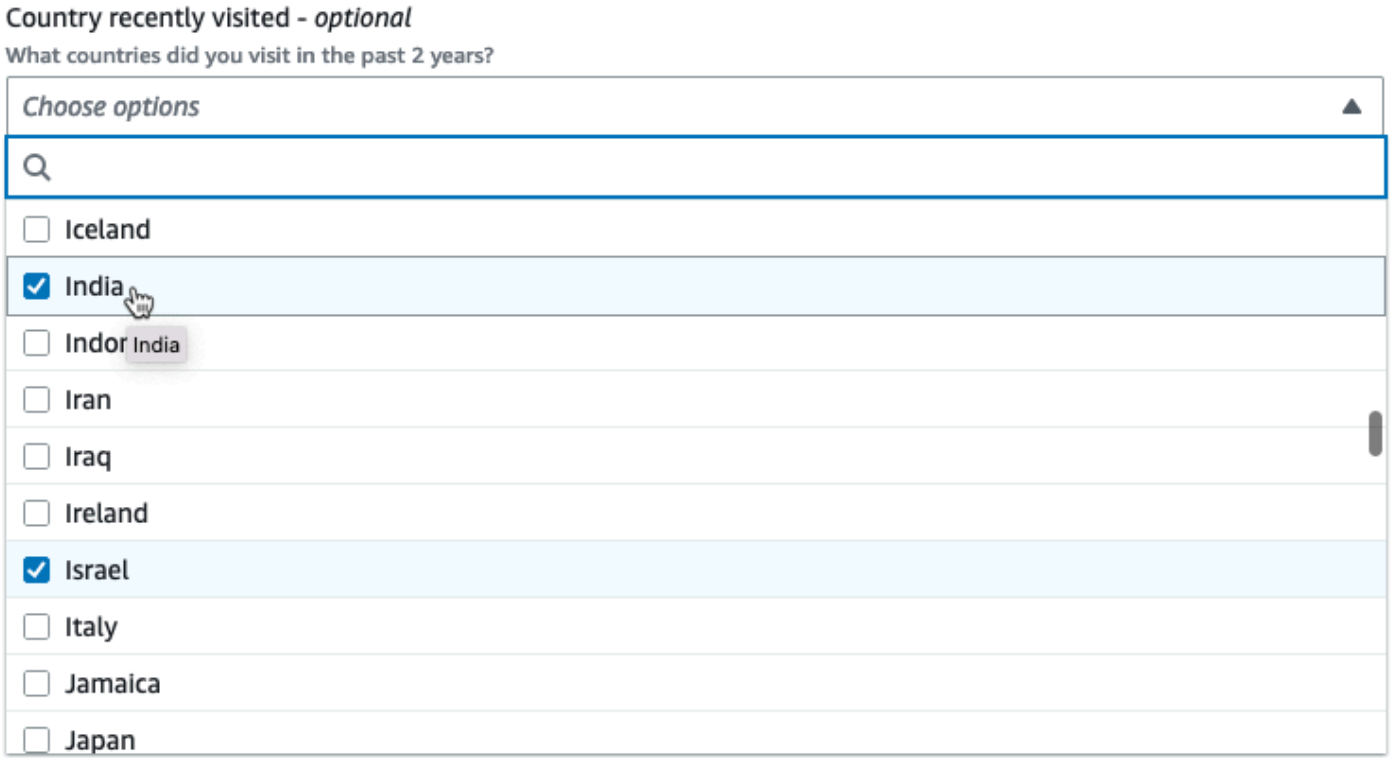
- Wenn `type` gleich `bool` ist, wird ein Kontrollkästchen angezeigt.



- Wenn `type` gleich `str` ist und `listOptions` angegeben wird, wird eine einzelne Auswahlliste angezeigt.



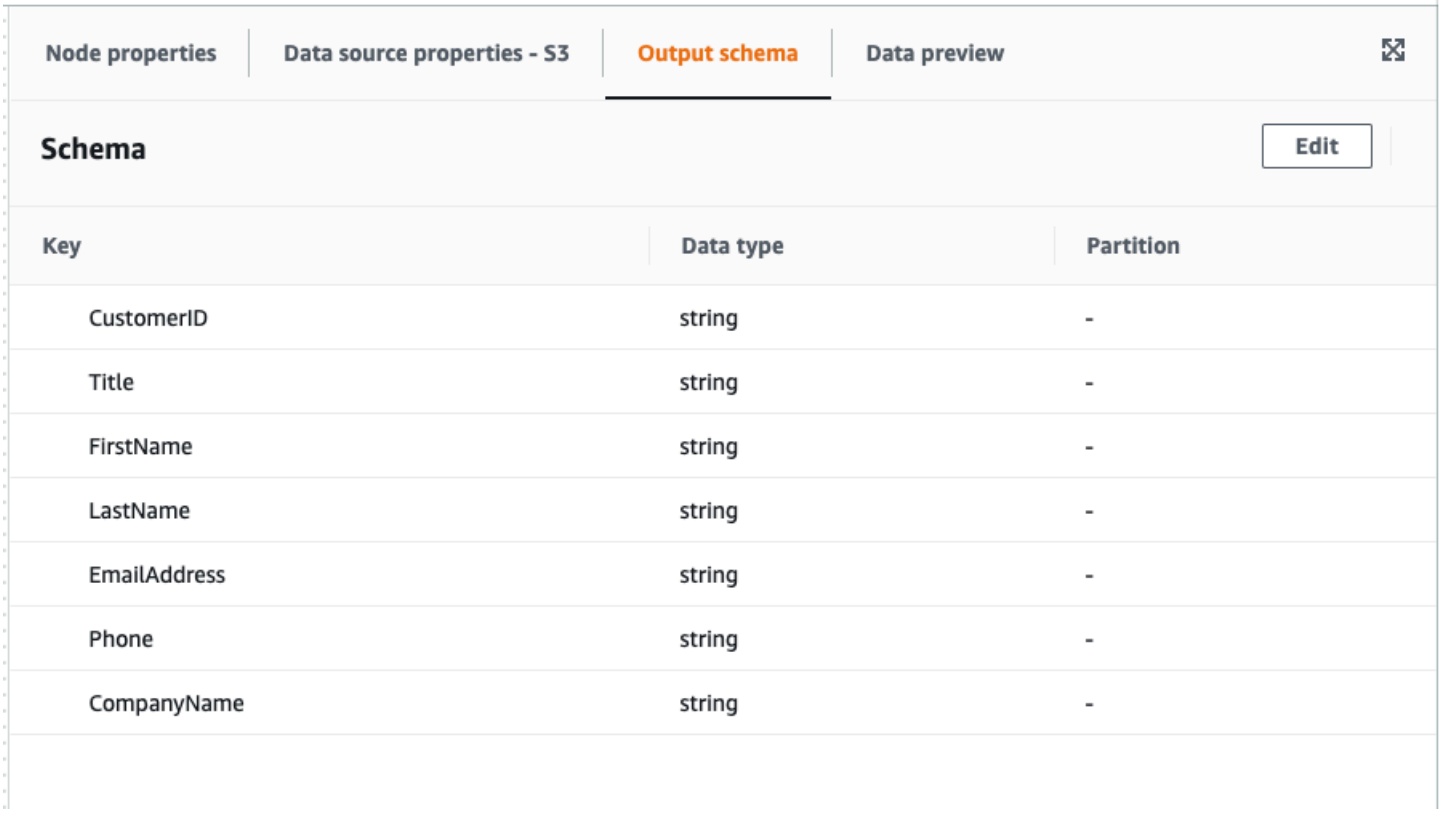
- Wenn `type` gleich `list` ist und `listOptions` und `listType` angegeben sind, wird eine Mehrfachauswahlliste angezeigt.



Anzeige einer Spaltenauswahl als Parameter

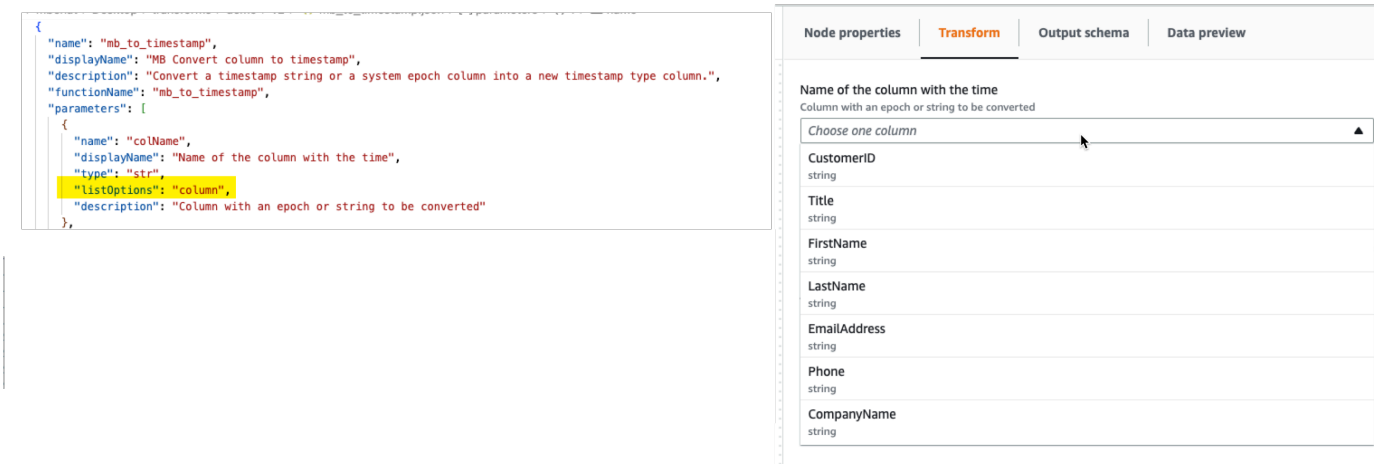
Wenn die Konfiguration erfordert, dass der Benutzer eine Spalte aus dem Schema auswählt, können Sie eine Spaltenauswahl anzeigen, sodass der Benutzer den Spaltennamen nicht eingeben muss. Indem Sie das `listOptions` Feld auf „Spalte“ setzen, AWS Glue Studio zeigt dynamisch eine Spaltenauswahl an, die auf dem Ausgabeschema des übergeordneten Knotens basiert. AWS Glue Studio kann entweder eine einzelne oder mehrere Spaltenauswahl anzeigen.

Im folgenden Beispiel wird das Schema verwendet:



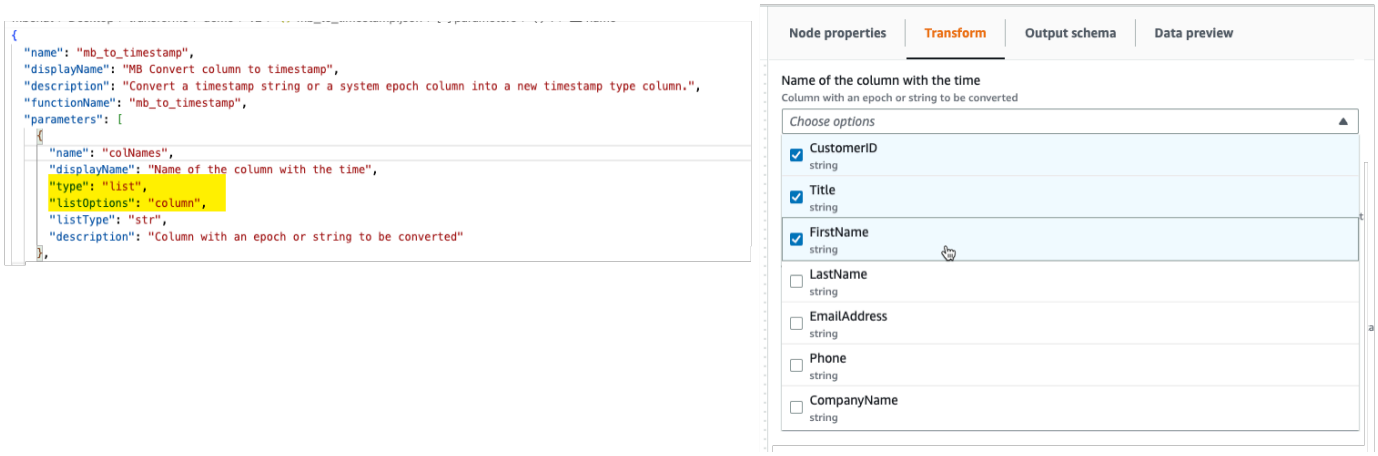
So definieren Sie Ihren benutzerdefinierten visuellen Transformationsparameter für die Anzeige einer einzelnen Spalte:

1. Legen Sie in Ihrer JSON-Datei für das `parameters`-Objekt den `listOptions`-Wert auf „column“ fest. Auf diese Weise kann ein Benutzer eine Spalte aus einer Auswahlliste in auswählen AWS Glue Studio.



2. Sie können auch die Auswahl mehrerer Spalten zulassen, indem Sie den Parameter wie folgt definieren:

   - `listOptions: "column"`
   - `type: "list"`



Schritt 2. Implementieren der Transformationslogik

> **Note**
>
> Benutzerdefinierte visuelle Transformationen unterstützen nur Python-Skripte. Scala wird nicht unterstützt.

Um den Code hinzuzufügen, der die in der .json-Konfigurationsdatei definierte Funktion implementiert, wird empfohlen, die Python-Datei am selben Ort wie die .json-Datei zu platzieren, mit demselben Namen, aber mit der Erweiterung „.py“. AWS Glue Studio verbindet automatisch die Dateien.json und .py, sodass Sie den Pfad der Python-Datei nicht in der Konfigurationsdatei angeben müssen.

Fügen Sie in der Python-Datei die deklarierte Funktion mit den konfigurierten benannten Parametern hinzu und registrieren Sie sie zur Verwendung in `DynamicFrame`. Das Folgende ist ein Beispiel für eine Python-Datei:

```
from awsglue import DynamicFrame

# self refers to the DynamicFrame to transform,
# the parameter names must match the ones defined in the config
# if it's optional, need to provide a default value
def myTransform(self, email, phone, age=None, gender="",
                      country="", promotion=False):
   resulting_dynf = # do some transformation on self
   return resulting_dynf

DynamicFrame.myTransform = myTransform
```

Es wird empfohlen, eine zu verwenden AWS Glue Notizbuch für den schnellsten Weg, den Python-Code zu entwickeln und zu testen. Weitere Informationen finden Sie unter Erste Schritte mit Notizbüchern in AWS Glue Studio.

Um zu veranschaulichen, wie die Transformationslogik implementiert wird, ist die benutzerdefinierte visuelle Transformation im folgenden Beispiel eine Transformation zum Filtern eingehender Daten. Dabei werden nur die Daten berücksichtigt, die sich auf einen bestimmten US-Bundesstaat beziehen. Die JSON-Datei enthält den Parameter für `functionName` als `custom_filter_state` und zwei Argumente („state“ und „colName“ mit Typ „str“).

Die JSON-Beispielkonfigurationsdatei lautet:

```
{
"name": "custom_filter_state",
"displayName": "Filter State",
"description": "A simple example to filter the data to keep only the state indicated.",
"functionName": "custom_filter_state",
"parameters": [
```

```
    {
     "name": "colName",
     "displayName": "Column name",
     "type": "str",
     "description": "Name of the column in the data that holds the state postal code"
    },
    {
     "name": "state",
     "displayName": "State postal code",
     "type": "str",
     "description": "The postal code of the state whole rows to keep"
    }
   ]
}
```

So implementieren Sie das Begleitskript in Python

1. Starten Sie ein AWS Glue notieren Sie und führen Sie die erste Zelle aus, die für den Start der Sitzung vorgesehen ist. Durch Ausführen der Anfangszelle werden die erforderlichen Basiskomponenten erstellt.
2. Erstellen Sie eine Funktion, die die im Beispiel beschriebene Filterung durchführt, und registrieren Sie sie auf `DynamicFrame`. Kopieren Sie den folgenden Code und fügen Sie ihn in eine Zelle im AWS Glue Notizbuch.

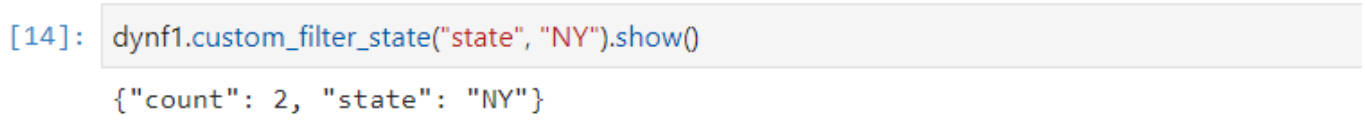
   ```
   from awsglue import DynamicFrame

   def custom_filter_state(self, colName, state):
       return self.filter(lambda row: row[colName] == state)

   DynamicFrame.custom_filter_state = custom_filter_state
   ```

3. Erstellen oder laden Sie Beispieldaten, um den Code in derselben Zelle oder in einer neuen Zelle zu testen. Wenn Sie die Beispieldaten in einer neuen Zelle hinzufügen, vergessen Sie nicht, die Zelle auszuführen. Zum Beispiel:

   ```
   # A few of rows of sample data to test
   data_sample = [
       {"state": "CA", "count": 4},
       {"state": "NY", "count": 2},
   ```

```
    {"state": "WA", "count": 3}
]
df1 = glueContext.sparkSession.sparkContext.parallelize(data_sample).toDF()
dynf1 = DynamicFrame.fromDF(df1, glueContext, None)
```

4. Führen Sie einen Test durch, um den „custom_filter_state“ mit verschiedenen Argumenten zu validieren:



5. Nachdem Sie mehrere Tests ausgeführt haben, speichern Sie den Code mit der PY-Erweiterung und benennen Sie die PY-Datei mit einem Namen, der den Namen der JSON-Datei widerspiegelt. Die PY- und JSON-Dateien sollten sich im selben Transformationsordner befinden.

   Kopieren Sie den folgenden Code, fügen Sie ihn in eine Datei ein und benennen Sie ihn mit einer PY-Dateierweiterung um.

```
from awsglue import DynamicFrame

def custom_filter_state(self, colName, state):
    return self.filter(lambda row: row[colName] == state)

DynamicFrame.custom_filter_state = custom_filter_state
```

6. In AWS Glue Studio, öffnen Sie einen Visual-Job und fügen Sie die Transformation dem Job hinzu, indem Sie sie aus der Liste der verfügbaren Transformationen auswählen.

   Um diese Transformation in einem Python-Skriptcode wiederzuverwenden, fügen Sie den Amazon-S3-Pfad zur PY-Datei im Auftrag unter „Referenzierter Dateipfad“ hinzu und importieren Sie im Skript den Namen der Python-Datei (ohne die Erweiterung), indem Sie ihn am Anfang der Datei hinzufügen. Zum Beispiel: `import` <Name der Datei (ohne die Erweiterung)>

Schritt 3. Validierung und Problembehebung benutzerdefinierter visueller Transformationen in AWS Glue Studio

AWS Glue Studio validiert die JSON-Konfigurationsdatei, bevor benutzerdefinierte visuelle Transformationen geladen werden AWS Glue Studio. Die Validierung beinhaltet:

- Vorhandensein von Pflichtfeldern
- JSON-Formatvalidierung
- Falsche oder ungültige Parameter
- Vorhandensein sowohl der PY- als auch der JSON-Dateien im selben Amazon-S3-Pfad
- Übereinstimmende Dateinamen für PY und JSON

Wenn die Validierung erfolgreich ist, wird die Transformation in der Liste der verfügbaren Actions (Aktionen) im visuellen Editor aufgeführt. Wenn ein benutzerdefiniertes Symbol bereitgestellt wurde, sollte es neben der Aktion sichtbar sein.

Wenn die Validierung fehlschlägt, AWS Glue Studio lädt die benutzerdefinierte visuelle Transformation nicht.

Schritt 4. Aktualisieren von benutzerdefinierten visuellen Transformationen nach Bedarf

Nach der Erstellung und Verwendung kann das Transformationsskript aktualisiert werden, solange die Transformation der entsprechenden JSON-Definition folgt:

- Der Name, der bei der Zuweisung verwendet wird, entspricht zu DynamicFrame viel dem JSON`functionName`.
- Die Funktionsargumente müssen in der JSON-Datei wie unter Schritt 1. Erstellen einer JSON-Konfigurationsdatei beschrieben definiert werden.
- Der Amazon-S3-Pfad der Python-Datei darf sich nicht ändern, da die Aufträge direkt davon abhängen.

> **Note**
>
> Wenn Aktualisierungen vorgenommen werden müssen, stellen Sie sicher, dass das Skript und die JSON-Datei konsistent aktualisiert werden und alle visuellen Aufträge mit der

neuen Transformation erneut korrekt gespeichert werden. Wenn visuelle Aufträge nach den Aktualisierungen nicht gespeichert werden, werden die Aktualisierungen nicht angewendet und validiert. Wenn die Python-Skriptdatei umbenannt oder nicht neben der JSON-Datei abgelegt wird, müssen Sie den vollständigen Pfad in der JSON-Datei angeben.

Benutzerdefiniertes Symbol

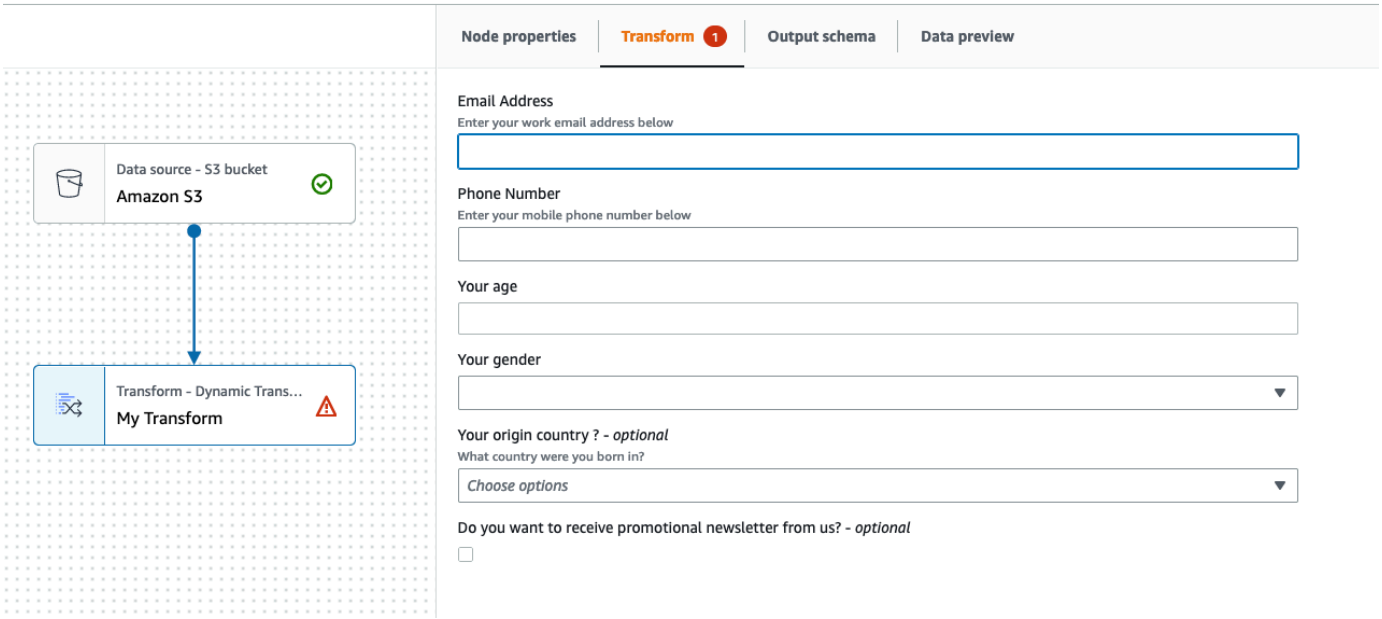
Wenn Sie feststellen, dass die Standardeinstellung für Ihre Aktion keine visuelle Unterscheidung im Rahmen Ihres Workflows zulässt, können Sie ein benutzerdefiniertes Symbol bereitstellen, wie in the section called “ Erste Schritte mit benutzerdefinierten visuellen Transformationen ” beschrieben. Sie können das Symbol aktualisieren, indem Sie die entsprechende SVG-Datei aktualisieren, die in Amazon S3 gehostet wird.

Um optimale Ergebnisse zu erzielen, entwerfen Sie Ihr Image so, dass es mit 32 x 32 Pixel angezeigt werden kann, und befolgen Sie dabei die Richtlinien des Cloudscape Design System. Weitere Informationen zu Cloudscape-Richtlinien finden Sie in der Cloudscape-Dokumentation

Schritt 5. Verwenden Sie benutzerdefinierte visuelle Transformationen in AWS Glue Studio

Um eine benutzerdefinierte visuelle Transformation zu verwenden AWS Glue Studio, laden Sie die Konfiguration und die Quelldateien hoch und wählen dann die Transformation aus dem Aktionsmenü aus. Alle Parameter, die Werte oder Eingaben benötigen, stehen Ihnen auf der Registerkarte Transform (Transformieren) zur Verfügung.

1. Laden Sie die beiden Dateien (Python-Quelldatei und JSON-Konfigurationsdatei) in den Ordner für Amazon-S3-Komponenten hoch, in dem die Auftragsskripte gespeichert sind. Standardmäßig AWS Glue ruft alle JSON-Dateien aus dem Ordner /transforms innerhalb desselben Amazon S3 S3-Buckets ab.
2. Wählen Sie im Menü Action (Aktion) die benutzerdefinierte visuelle Transformation aus. Es wird mit der Transformation `displayName` oder dem Namen benannt, den Sie in der JSON-Konfigurationsdatei angegeben haben.
3. Geben Sie Werte für alle Parameter ein, die in der Konfigurationsdatei konfiguriert wurden.



Verwendungsbeispiele

Das Folgende ist ein Beispiel aller möglichen Parameter in einer JSON-Konfigurationsdatei.

```
{
  "name": "MyTransform",
  "displayName": "My Transform",
  "description": "This transform description will be displayed in UI",
  "functionName": "myTransform",
  "parameters": [
      {
      "name": "email",
      "displayName": "Email Address",
      "type": "str",
      "description": "Enter your work email address below",
      "validationType": "RegularExpression",
      "validationRule": "^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w{2,3})+$",
      "validationMessage": "Please enter a valid email address"
    },
    {
      "name": "phone",
      "displayName": "Phone Number",
      "type": "str",
      "description": "Enter your mobile phone number below",
      "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
      "validationMessage": "Please enter a valid US number"
```

```
    },
    {
      "name": "age",
      "displayName": "Your age",
      "type": "int",
      "isOptional": true
    },
    {
      "name": "gender",
      "displayName": "Your gender",
      "type": "str",
      "listOptions": [
            {"label": "Male", "value": "male"},
            {"label": "Female", "value": "female"},
            {"label": "Other", "value": "other"}
        ],
      "isOptional": true
    },
    {
      "name": "country",
      "displayName": "Your origin country ?",
      "type": "list",
      "listOptions": "Afghanistan,Albania,Algeria,American
 Samoa,Andorra,Angola,Anguilla,Antarctica,Antigua and
 Barbuda,Argentina,Armenia,Aruba,Australia,Austria,Azerbaijan,Bahamas,Bahrain,Bangladesh,Barbad
 and Herzegovina,Botswana,Bouvet Island,Brazil,British Indian Ocean Territory,Brunei
 Darussalam,Bulgaria,Burkina Faso,Burundi,Cambodia,Cameroon,Canada,Cape
 Verde,Cayman Islands,Central African Republic,Chad,Chile,China,Christmas
 Island,Cocos (Keeling Islands),Colombia,Comoros,Congo,Cook Islands,Costa
 Rica,Cote D'Ivoire (Ivory Coast),Croatia (Hrvatska,Cuba,Cyprus,Czech
 Republic,Denmark,Djibouti,Dominica,Dominican Republic,East Timor,Ecuador,Egypt,El
 Salvador,Equatorial Guinea,Eritrea,Estonia,Ethiopia,Falkland Islands (Malvinas),Faroe
 Islands,Fiji,Finland,France,France,Metropolitan,French Guiana,French Polynesia,French
 Southern
 Territories,Gabon,Gambia,Georgia,Germany,Ghana,Gibraltar,Greece,Greenland,Grenada,Guadeloupe,G
Bissau,Guyana,Haiti,Heard and McDonald Islands,Honduras,Hong
 Kong,Hungary,Iceland,India,Indonesia,Iran,Iraq,Ireland,Israel,Italy,Jamaica,Japan,Jordan,Kazak
 (North),Korea
 (South),Kuwait,Kyrgyzstan,Laos,Latvia,Lebanon,Lesotho,Liberia,Libya,Liechtenstein,Lithuania,Lu
 Islands,Martinique,Mauritania,Mauritius,Mayotte,Mexico,Micronesia,Moldova,Monaco,Mongolia,Mont
 Antilles,New Caledonia,New Zealand,Nicaragua,Niger,Nigeria,Niue,Norfolk
 Island,Northern Mariana Islands,Norway,Oman,Pakistan,Palau,Panama,Papua
 New Guinea,Paraguay,Peru,Philippines,Pitcairn,Poland,Portugal,Puerto
 Rico,Qatar,Reunion,Romania,Russian Federation,Rwanda,Saint Kitts and Nevis,Saint
```

```
 Lucia,Saint Vincent and The Grenadines,Samoa,San Marino,Sao Tome and Principe,Saudi
 Arabia,Senegal,Seychelles,Sierra Leone,Singapore,Slovak Republic,Slovenia,Solomon
 Islands,Somalia,South Africa,S. Georgia and S. Sandwich Isls.,Spain,Sri
 Lanka,St. Helena,St. Pierre and Miquelon,Sudan,Suriname,Svalbard and Jan Mayen
 Islands,Swaziland,Sweden,Switzerland,Syria,Tajikistan,Tanzania,Thailand,Togo,Tokelau,Tonga,Tri
 and Tobago,Tunisia,Turkey,Turkmenistan,Turks and Caicos
 Islands,Tuvalu,Uganda,Ukraine,United Arab Emirates,United Kingdom
 (Britain / UK),United States of America (USA),US Minor Outlying
 Islands,Uruguay,Uzbekistan,Vanuatu,Vatican City State (Holy See),Venezuela,Viet
 Nam,Virgin Islands (British),Virgin Islands (US),Wallis and Futuna Islands,Western
 Sahara,Yemen,Yugoslavia,Zaire,Zambia,Zimbabwe",
      "description": "What country were you born in?",
      "listType": "str",
      "isOptional": true
    },
    {
      "name": "promotion",
      "displayName": "Do you want to receive promotional newsletter from us?",
      "type": "bool",
      "isOptional": true
    }
  ]
}
```

Beispiele für benutzerdefinierte visuelle Skripte

In den folgenden Beispielen werden äquivalente Transformationen durchgeführt. Das zweite Beispiel (SparkSQL) ist jedoch das klarste und effizienteste, gefolgt von der Pandas-UDF und schließlich dem Low-Level-Mapping im ersten Beispiel. Folgendes ist ein vollständiges Beispiel für eine einfache Transformation zum Addieren von zwei Spalten:

```
from awsglue import DynamicFrame

# You can have other auxiliary variables, functions or classes on this file, it won't
 affect the runtime
def record_sum(rec, col1, col2, resultCol):
    rec[resultCol] = rec[col1] + rec[col2]
    return rec


# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
```

```
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    # The mapping will alter the columns order, which could be important
    fields = [field.name for field in self.schema()]
    if resultCol not in fields:
        # If it's a new column put it at the end
        fields.append(resultCol)
    return self.map(lambda record: record_sum(record, col1, col2,
 resultCol)).select_fields(paths=fields)


# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Das folgende Beispiel ist eine äquivalente Transformation, die die SparkSQL-API nutzt.

```
from awsglue import DynamicFrame

# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, df[col1] + df[col2]) # This is the conversion logic
        , self.glue_ctx, self.name)


# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Das folgende Beispiel verwendet dieselben Transformationen, verwendet jedoch eine Pandas-UDF, die effizienter ist als die Verwendung einer einfachen UDF. Weitere Informationen zum Schreiben von Pandas UDFs finden Sie in der Apache Spark-SQL-Dokumentation.

```
from awsglue import DynamicFrame
import pandas as pd
from pyspark.sql.functions import pandas_udf
```

```
# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    @pandas_udf("integer")  # We need to declare the type of the result column
    def add_columns(value1: pd.Series, value2: pd.Series) # pd.Series:
        return value1 + value2

    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, add_columns(col1, col2)) # This is the conversion
 logic
        , self.glue_ctx, self.name)

# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Video

Das folgende Video bietet eine Einführung in visuelle benutzerdefinierte Transformationen und zeigt, wie diese verwendet werden.

Verwenden von Data Lake-Frameworks mit AWS Glue Studio

Übersicht

Open-Source-Data-Lake-Frameworks vereinfachen die inkrementelle Datenverarbeitung für Dateien, die in Data Lakes gespeichert sind, die auf Amazon S3 basieren. AWS Glue 3.0 und höher unterstützt die folgenden Open-Source-Data-Lake-Speicher-Frameworks:

- Apache Hudi
- Linux Foundation Delta Lake
- Apache Iceberg

Ab AWS Glue 4.0, AWS Glue bietet native Unterstützung für diese Frameworks, sodass Sie Daten, die Sie in Amazon S3 speichern, transaktionskonsistent lesen und schreiben können. Es ist nicht erforderlich, einen separaten Konnektor zu installieren oder zusätzliche Konfigurationsschritte durchzuführen, um diese Frameworks in zu verwenden AWS Glue Jobs.

Data Lake-Frameworks können als Quelle oder Ziel innerhalb verwendet werden AWS Glue Studio durch Spark Script Editor-Jobs. Weitere Informationen zur Verwendung von Apache Hudi, Apache Iceberg und Delta Lake finden Sie unter: Verwenden von Data Lake-Frameworks mit AWS Glue ETL-Jobs.

Open-Table-Formate aus einer AWS Glue Streaming-Quelle erstellen

AWS Glue Streaming-ETL-Jobs verbrauchen kontinuierlich Daten aus Streaming-Quellen, bereinigen und transformieren die Daten während der Übertragung und stellen sie innerhalb von Sekunden für Analysen zur Verfügung.

AWS bietet eine breite Auswahl an Dienstleistungen zur Unterstützung Ihrer Bedürfnisse. Ein Datenbankreplikationsservice wie AWS Database Migration Service kann die Daten aus Ihren Quellsystemen nach Amazon S3 replizieren, das üblicherweise die Speicherebene des Data Lake hostet. Es ist zwar einfach, Updates in einem relationalen Datenbankmanagementsystem (RDBMS) anzuwenden, das eine Online-Quellanwendung unterstützt, aber es ist kompliziert, diesen CDC-Prozess auf Ihre Data Lakes anzuwenden. Die Open-Source-Frameworks für Datenmanagement vereinfachen die inkrementelle Datenverarbeitung und die Entwicklung von Datenpipelines und sind eine gute Option zur Lösung dieses Problems.

Weitere Informationen finden Sie unter:

- Erstellen Sie mithilfe von Streaming einen auf Apache HUDI basierenden near-real-time transaktionalen Data Lake AWS Glue
- Erstellen Sie einen Data Lake für Apache Iceberg in Echtzeit, der an die DSGVO angepasst ist

Verwenden des Hudi-Frameworks in AWS Glue Studio

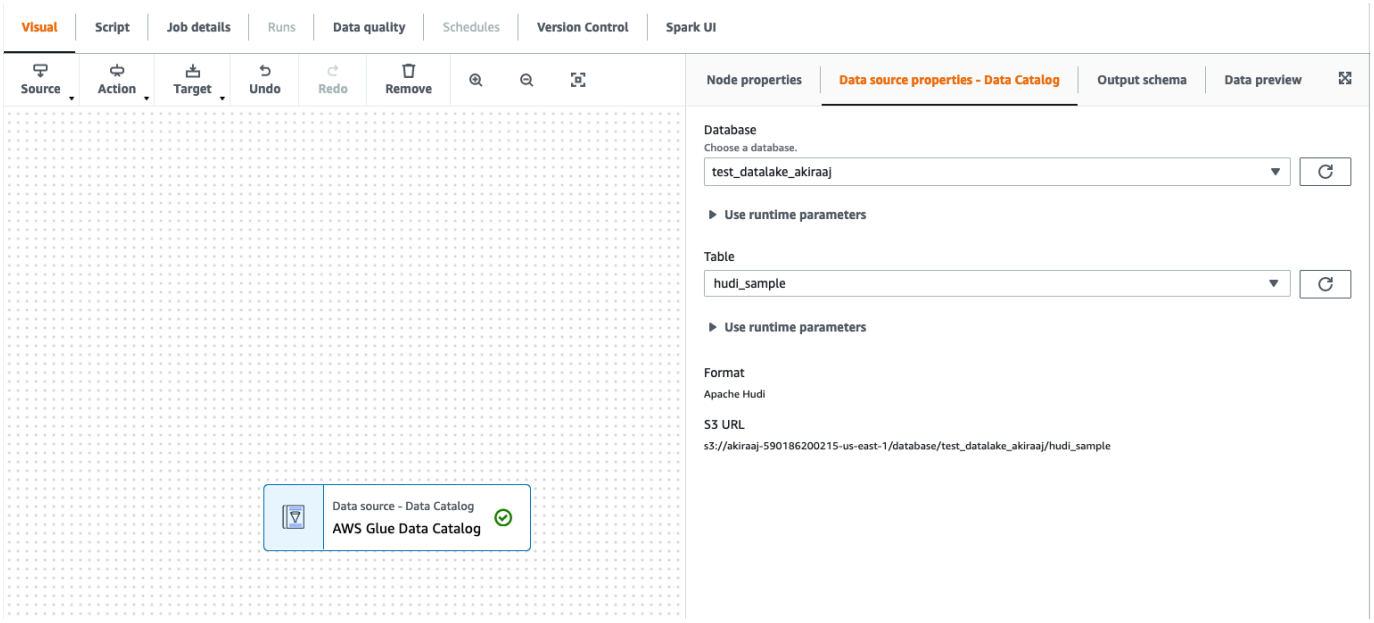
Beim Erstellen oder Bearbeiten eines Jobs AWS Glue Studio fügt automatisch die entsprechenden Hudi-Bibliotheken für Sie hinzu, abhängig von der Version von AWS Glue du verwendest. Weitere Informationen finden Sie unter Verwenden des Hudi-Frameworks in AWS Glue.

Verwendung des Apache-Hudi-Frameworks in Datenquellen des Datenkatalogs

So fügen Sie einem Auftrag ein Hudi-Datenquellenformat hinzu:

1. Wählen Sie im Menü Quelle die Option AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.

3. AWS Glue Studio zeigt den Formattyp als Apache Hudi und die Amazon S3 S3-URL an.



Verwenden des Hudi-Frameworks in Amazon-S3-Datenquellen

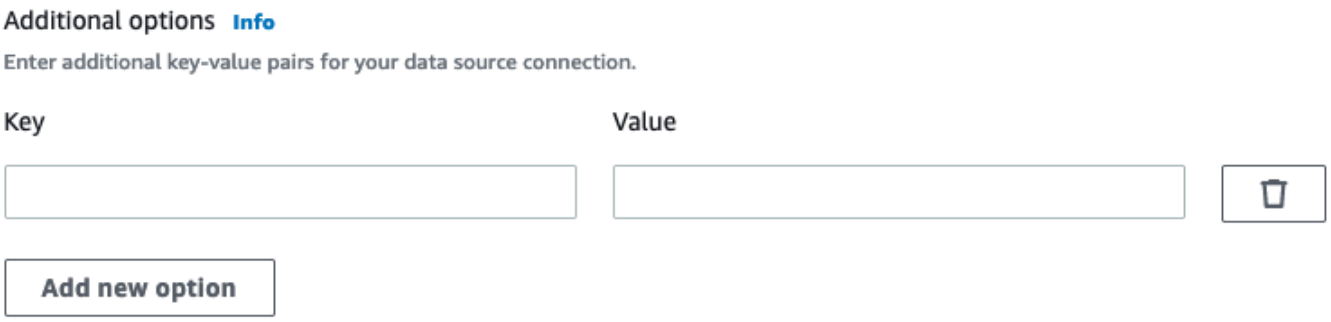
1. Wählen Sie im Menü Quelle Amazon S3.
2. Wenn Sie die Datenkatalog-Tabelle als Amazon-S3-Quelltyp wählen, wählen Sie eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt das Format als Apache Hudi und die Amazon S3 S3-URL an.
4. Wenn Sie den Amazon-S3-Standort als Amazon-S3-Quelltyp wählen, wählen Sie die Amazon-S3-URL aus, indem Sie auf Amazon S3 durchsuchen klicken.
5. Wählen Sie unter Datenformat die Option Apache-Hudi aus.

> **Note**
>
> Wenn AWS Glue Studio kann das Schema nicht aus dem ausgewählten Amazon S3 S3-Ordner oder der ausgewählten Datei ableiten. Wählen Sie Zusätzliche Optionen, um einen neuen Ordner oder eine neue Datei auszuwählen.
> Wählen Sie unter Zusätzliche Optionen unter Schema-Inferenz aus den folgenden Optionen:
>
> - Lass AWS Glue Studio automatisch eine Beispieldatei auswählen — AWS Glue Studio wählt eine Beispieldatei am Amazon S3 S3-Speicherort aus, sodass das Schema

abgeleitet werden kann. Im Feld Datei mit automatischem Sampling können Sie die Datei anzeigen, die automatisch ausgewählt wurde.

- Wählen Sie eine Beispieldatei aus Amazon S3 – wählen Sie die Amazon-S3-Datei aus, die Sie verwenden möchten, indem Sie auf Amazon S3 durchsuchen klicken.

6. Klicken Sie auf Schema ableiten. Sie können dann das Ausgabeschema anzeigen, indem Sie auf die Registerkarte Ausgabeschema klicken.
7. Wählen Sie Zusätzliche Optionen, um ein Schlüssel-Wert-Paar einzugeben.



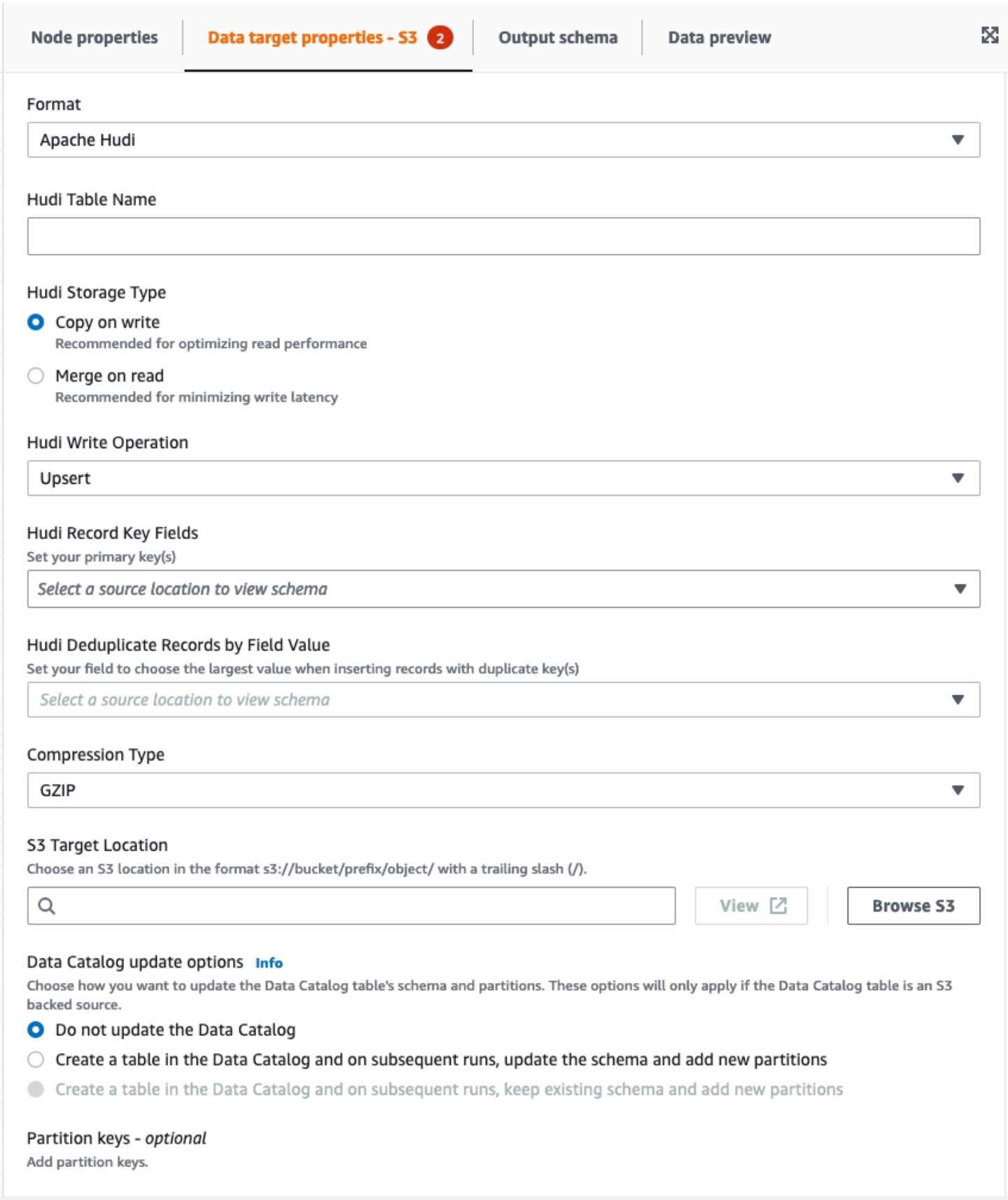
Verwendung des Apache-Hudi-Frameworks in Datenzielen

Verwendung des Apache Hudi-Frameworks in Datenkatalog-Datenzielen

1. Wählen Sie im Menü Target AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt den Formattyp als Apache Hudi und die Amazon S3 S3-URL an.

Verwendung des Apache-Hudi-Frameworks in Amazon-S3-Datenzielen

Geben Sie Werte ein oder wählen Sie aus den verfügbaren Optionen, um das Apache-Hudi-Format zu konfigurieren. Weitere Informationen zu Apache Hudi finden Sie in der Apache-Hudi-Dokumentation.


Node properties
Data target properties - S3 2
Output schema
Data preview
Format
Apache Hudi
Hudi Table Name
Hudi Storage Type
Copy on write
Recommended for optimizing read performance
Merge on read
Recommended for minimizing write latency
Hudi Write Operation
Upsert
Hudi Record Key Fields
Set your primary key(s)
Select a source location to view schema
Hudi Deduplicate Records by Field Value
Set your field to choose the largest value when inserting records with duplicate key(s)
Select a source location to view schema
Compression Type
GZIP
S3 Target Location
Choose an S3 location in the format s3://bucket/prefix/object/ with a trailing slash (/).
View
Browse S3
Data Catalog update options Info
Choose how you want to update the Data Catalog table's schema and partitions. These options will only apply if the Data Catalog table is an S3 backed source.
Do not update the Data Catalog
Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions
Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions
Partition keys - optional
Add partition keys.

- Hudi-Tabellenname – das ist der Name Ihrer Hudi-Tabelle.
- Hudi Speichertyp – wählen Sie aus zwei Optionen:
  - Kopieren beim Schreiben – empfohlen für die Optimierung der Leseleistung. Das ist der Standardspeichertyp für Hudi. Jede Aktualisierung erstellt während eines Schreibvorgangs eine neue Version der Dateien.
  - Beim Lesen zusammenführen – empfohlen, um die Schreiblatenz zu minimieren. Updates werden in zeilenbasierten Delta-Dateien protokolliert und nach Bedarf komprimiert, um neue Versionen der Spaltendateien zu erstellen.
- Hudi-Schreibvorgang – wählen aus den folgenden Optionen:
  - Upsert – Dies ist die Standardoperation, bei der die Eingabedatensätze zuerst als Einfügungen oder Aktualisierungen gekennzeichnet werden, indem im Index nachgeschlagen wird. Wird empfohlen, wenn Sie bestehende Daten aktualisieren.
  - Einfügen – Dadurch werden Datensätze eingefügt, aber nicht nach vorhandenen Datensätzen gesucht, was zu Duplikaten führen kann.
  - Massen-Einfügen – Dies fügt Datensätze ein und wird für große Datenmengen empfohlen.
- Hudi-Datensatz-Schlüsselfelder – verwenden Sie die Suchleiste, um nach primären Datensatzschlüsseln zu suchen und diese auszuwählen. Datensätze in Hudi werden durch einen Primärschlüssel identifiziert, der ein Paar aus Datensatzschlüssel und Partitionspfad ist, zu dem der Datensatz gehört.
- Hudi-Vorab-Kombinationsfeld – dies ist das Feld, das in Vorabkombination vor dem eigentlichen Schreiben verwendet wird. Wenn zwei Datensätze denselben Schlüsselwert haben, AWS Glue Studio wählt den mit dem größten Wert für das Präkombinationsfeld aus. Stellen Sie ein Feld ein, zu dem der inkrementelle Wert gehört (z. B. updated_at).
- Komprimierungstyp – Wählen Sie eine der Optionen für den Komprimierungstyp: Unkomprimiert, GZIP, LZO oder Snappy.
- Amazon-S3-Zielstandort – Wählen Sie den Amazon-S3-Zielstandort aus, indem Sie auf S3 durchsuchen klicken.
- Aktualisierungsoptionen aus dem Datenkatalog – wählen Sie aus den folgenden Optionen aus:
  - Do not update the Data Catalog (Data Catalog nicht aktualisieren): (Standard) Wählen Sie diese Option, wenn der Auftrag den Data Catalog nicht aktualisieren soll, selbst wenn sich das Schema ändert oder neue Partitionen hinzukommen.
  - Erstellen Sie eine Tabelle im Datenkatalog und aktualisieren Sie bei späteren Läufen das Schema und fügen Sie neue Partitionen hinzu: Wenn Sie diese Option wählen, erstellt der Auftrag die Tabelle im Datenkatalog bei der ersten Ausführung des Auftrags. Bei nachfolgenden

Ausführungen aktualisiert der Auftrag die Data-Catalog-Tabelle, wenn sich das Schema ändert oder neue Partitionen hinzugefügt werden.

Sie müssen auch eine Datenbank aus dem Data Catalog auswählen und einen Tabellennamen eingeben.

- Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions (Eine Tabelle im Data Catalog erstellen und bei folgenden Ausführungen das bestehende Schema beibehalten und neue Partitionen hinzufügen): Wenn Sie diese Option wählen, erstellt der Auftrag die Tabelle im Data Catalog bei der ersten Ausführung des Auftrags. Bei nachfolgenden Ausführungen fügt der Auftrag lediglich neue Partitionen in die Data-Catalog-Tabelle ein.

  Sie müssen auch eine Datenbank aus dem Data Catalog auswählen und einen Tabellennamen eingeben.

- Partition keys (Partitionsschlüssel): Wählen Sie aus, welche Spalten als Partitionierungsschlüssel in der Ausgabe verwendet werden sollen. Um weitere Partitionsschlüssel hinzuzufügen, wählen Sie Add a partition key (Partitionsschlüssel hinzufügen) aus.
- Zusätzliche Optionen – Geben Sie nach Bedarf ein Schlüssel-Wert-Paar ein.

Code wird generiert durch AWS Glue Studio

Wenn der Auftrag gespeichert wird, werden die folgenden Auftragsparameter zum Auftrag hinzugefügt, wenn eine Hudi-Quelle oder ein Hudi-Ziel erkannt wird:

- `--datalake-formats` – eine eindeutige Liste der Data Lake-Formate, die im visuellen Auftrag erkannt wurden (entweder direkt durch Auswahl eines „Formats“ oder indirekt durch Auswahl einer Katalogtabelle, die von einem Data Lake unterstützt wird).
- `--conf` – generiert auf der Grundlage des Werts von `--datalake-formats`. Wenn der Wert für beispielsweise 'hudi' `--datalake-formats` ist, AWS Glue generiert einen Wert von `spark.serializer=org.apache.spark.serializer.KryoSerializer –conf spark.sql.hive.convertMetastoreParquet=false` für diesen Parameter.

Überschreiben AWS Glue-bereitgestellte Bibliotheken

Um eine Version von Hudi zu verwenden, die AWS Glue nicht unterstützt, Sie können Ihre eigenen JAR-Dateien für die Hudi-Bibliothek angeben. So verwenden Sie eine JAR-Datei:

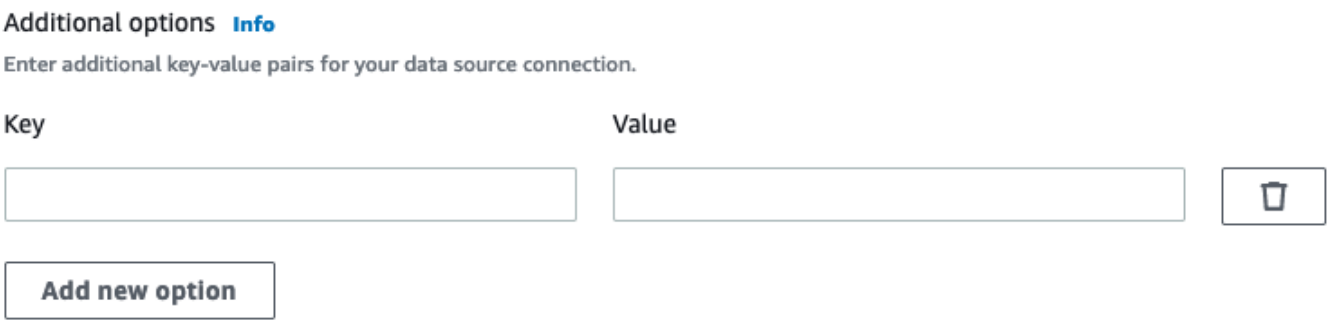
- Verwenden Sie den Auftragsparameter `--extra-jars`. Beispiel, `'--extra-jars': 's3pathtojarfile.jar'`. Weitere Informationen finden Sie unter AWS Glue Jobparameter.
- Schließen Sie `hudi` nicht als Wert für den Auftragsparameter `--datalake-formats` ein. Wenn Sie eine leere Zeichenfolge als Wert eingeben, wird sichergestellt, dass Ihnen keine Data Lake-Bibliotheken zur Verfügung gestellt werden von AWS Glue automatisch. Weitere Informationen finden Sie unter Verwenden des Hudi-Frameworks in AWS Glue.

Verwenden des Delta Lake-Frameworks in AWS Glue Studio

Verwendung des Delta-Lake-Frameworks in Datenquellen

Verwendung des Delta-Lake-Frameworks in Amazon-S3-Datenquellen

1. Wählen Sie im Menü Quelle Amazon S3.
2. Wenn Sie die Datenkatalog-Tabelle als Amazon-S3-Quelltyp wählen, wählen Sie eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt das Format als Delta Lake und die Amazon S3 S3-URL an.
4. Wählen Sie Zusätzliche Optionen, um ein Schlüssel-Wert-Paar einzugeben. Ein Schlüssel-Wert-Paar könnte beispielsweise sein: Schlüssel: timestampAsOf und Wert: 2023-02-24 14:16:18.



5. Wenn Sie den Amazon-S3-Standort als Amazon-S3-Quelltyp wählen, wählen Sie die Amazon-S3-URL aus, indem Sie auf Amazon S3 durchsuchen klicken.
6. Wählen Sie unter Datenformat die Option Delta Lake aus.

> **Note**
>
> Wenn AWS Glue Studio kann das Schema nicht aus dem ausgewählten Amazon S3 S3-Ordner oder der ausgewählten Datei ableiten. Wählen Sie Zusätzliche Optionen, um einen neuen Ordner oder eine neue Datei auszuwählen.
> Wählen Sie unter Zusätzliche Optionen unter Schema-Inferenz aus den folgenden Optionen:
>
> - Lass AWS Glue Studio automatisch eine Beispieldatei auswählen — AWS Glue Studio wählt eine Beispieldatei am Amazon S3 S3-Speicherort aus, sodass das Schema abgeleitet werden kann. Im Feld Datei mit automatischem Sampling können Sie die Datei anzeigen, die automatisch ausgewählt wurde.
> - Wählen Sie eine Beispieldatei aus Amazon S3 – wählen Sie die Amazon-S3-Datei aus, die Sie verwenden möchten, indem Sie auf Amazon S3 durchsuchen klicken.

7. Klicken Sie auf Schema ableiten. Sie können das Ausgabeschema dann anzeigen, indem Sie auf die Registerkarte Ausgabeschema klicken.

Verwendung des Delta-Lake-Frameworks in Datenkatalog-Datenquellen

1. Wählen Sie im Menü Quelle AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt den Formattyp als Delta Lake und die Amazon S3 S3-URL an.

> **Note**
>
> Wenn Ihre Delta Lake-Quelle nicht registriert ist als AWS Glue In der Datenkatalogtabelle haben Sie noch zwei Möglichkeiten:
>
> 1. Erstellen Sie eine AWS Glue Crawler für den Delta Lake-Datenspeicher. Weitere Informationen finden Sie unter So geben Sie Konfigurationsoptionen für einen Delta-Lake-Datenspeicher an.

2. Verwenden aus einer Amazon-S3-Datenquelle, um Ihre Delta-Lake-Datenquelle auszuwählen. Siehe Verwendung des Delta-Lake-Frameworks in Amazon-S3-Datenquellen .

Verwendung von Delta-Lake-Formaten in Datenzielen

Verwendung von Delta-Lake-Formaten in Datenkatalog-Datenzielen

1. Wählen Sie im Menü Ziel AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt den Formattyp als Delta Lake und die Amazon S3 S3-URL an.

Verwendung von Delta-Lake-Formaten in Amazon-S3-Datenquellen

Geben Sie Werte ein oder wählen Sie aus den verfügbaren Optionen, um das Delta–Lake-Format zu konfigurieren.

- Komprimierungstyp – Wählen Sie eine der Optionen für den Komprimierungstyp: Unkomprimiert oder Snappy.
- Amazon-S3-Zielstandort – Wählen Sie den Amazon-S3-Zielstandort aus, indem Sie auf S3 durchsuchen klicken.
- Aktualisierungsoptionen für den Datenkatalog – Die Aktualisierung des Datenkatalogs wird für dieses Format im visuellen Editor von Glue Studio nicht unterstützt.
  - Do not update the Data Catalog (Data Catalog nicht aktualisieren): (Standard) Wählen Sie diese Option, wenn der Auftrag den Data Catalog nicht aktualisieren soll, selbst wenn sich das Schema ändert oder neue Partitionen hinzukommen.
  - Um den Datenkatalog nach dem zu aktualisieren AWS Glue Ausführung eines Jobs, Ausführung oder Planung eines AWS Glue Crawler. Weitere Informationen finden Sie unter So geben Sie Konfigurationsoptionen für einen Delta-Lake-Datenspeicher an.
- Partitionsschlüssel: Wählen Sie aus, welche Spalten als Partitionsschlüssel in der Ausgabe verwendet werden sollen. Um weitere Partitionsschlüssel hinzuzufügen, wählen Sie Add a partition key (Partitionsschlüssel hinzufügen) aus.
- Wählen Sie Zusätzliche Optionen, um ein Schlüssel-Wert-Paar einzugeben. Ein Schlüssel-Wert-Paar könnte beispielsweise sein: Schlüssel: timestampAsOf und Wert: 2023-02-24 14:16:18.

Verwenden des Apache Iceberg-Frameworks in AWS Glue Studio

Verwendung des Apache-Iceberg-Frameworks in Datenzielen

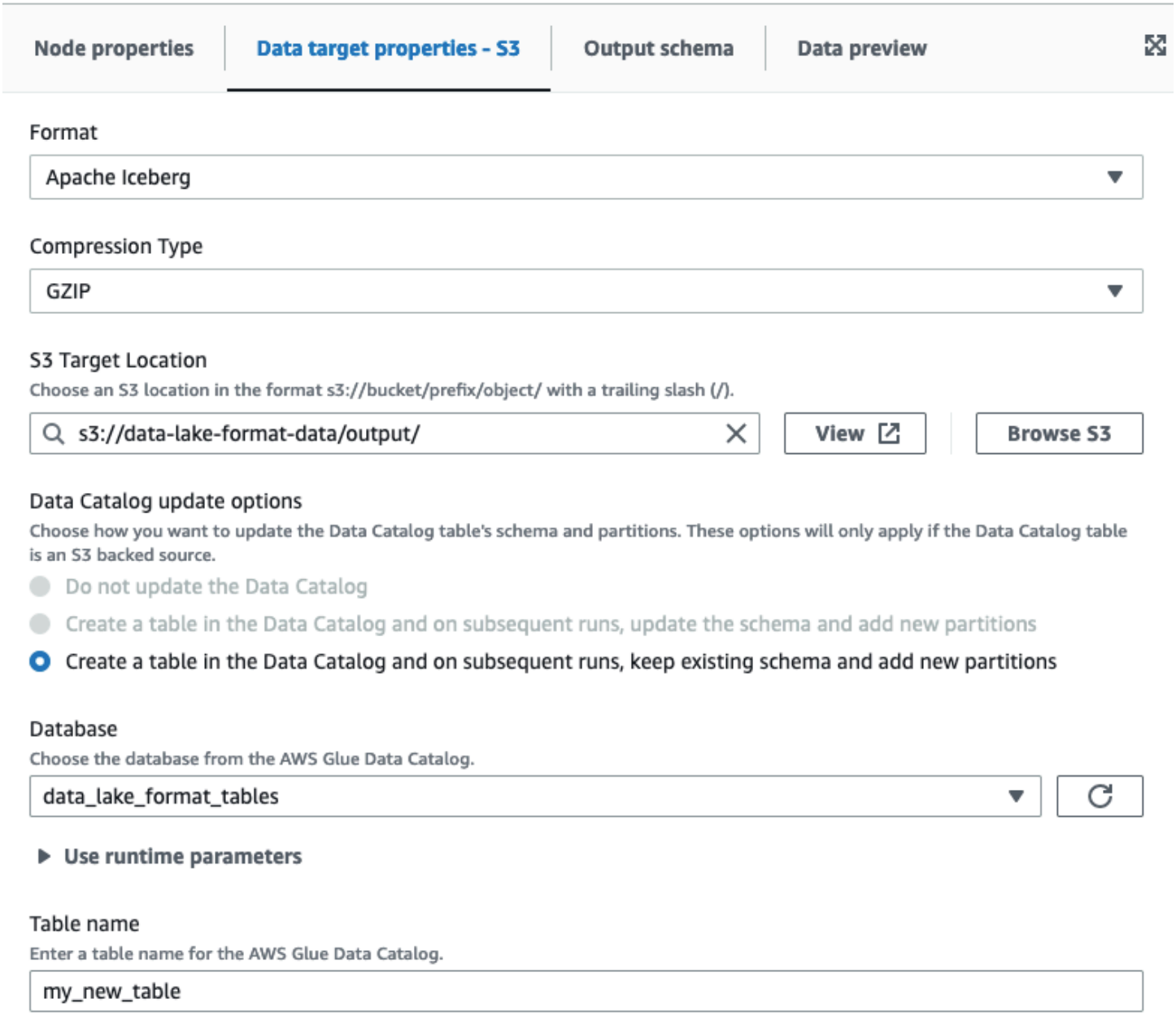
Verwendung des Apache-Iceberg-Frameworks in Data-Catalog-Datenzielen

1. Wählen Sie im Target-Menü AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt den Formattyp als Apache Iceberg und die Amazon S3 S3-URL an.

Verwendung des Apache-Iceberg-Frameworks in Amazon-S3-Datenzielen

Geben Sie Werte ein oder wählen Sie aus den verfügbaren Optionen aus, um das Apache-Iceberg-Format zu konfigurieren.

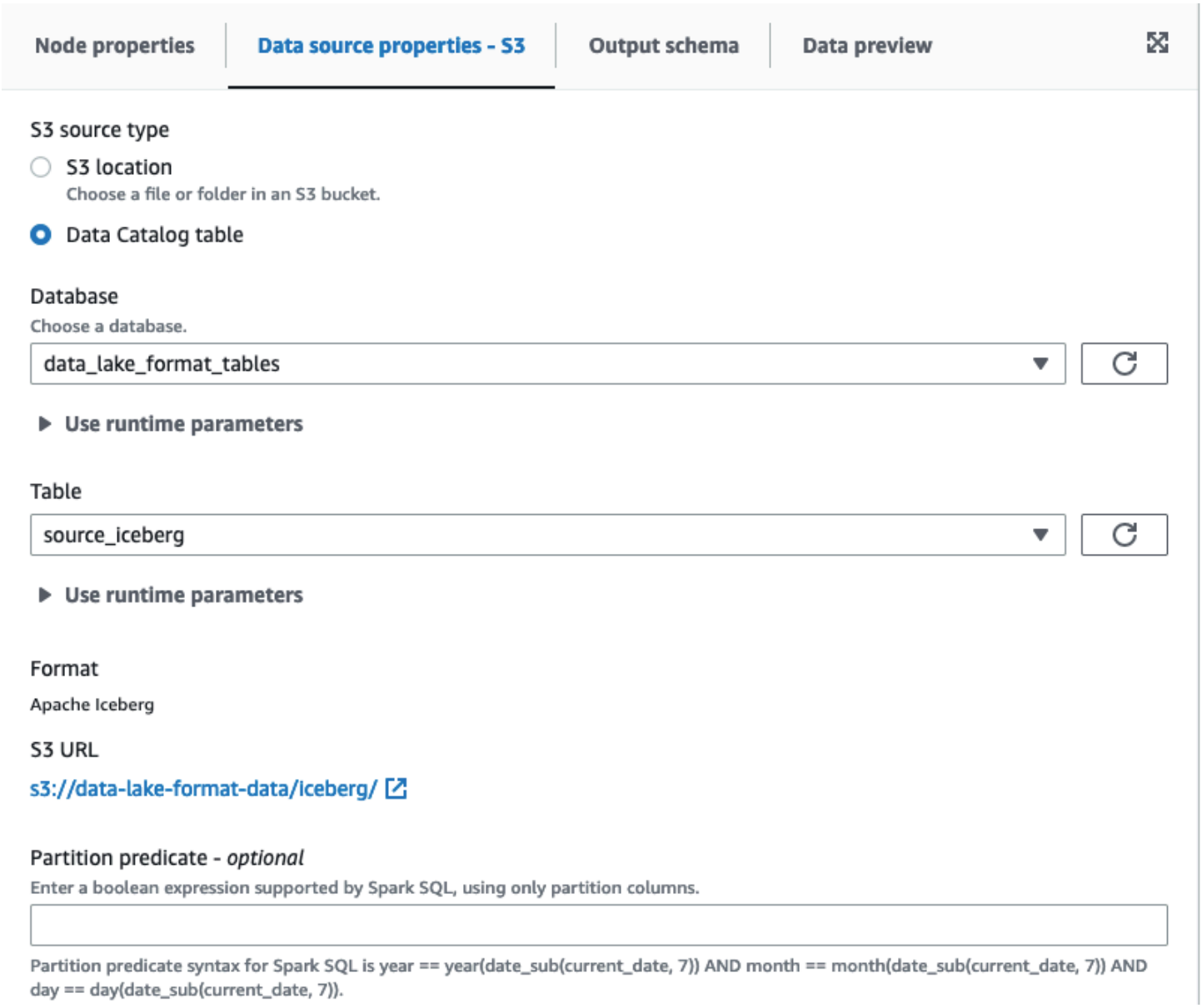
- Format – wählen Sie Apache Iceberg aus dem Dropdown-Menü.
- Amazon-S3-Zielspeicherort – Wählen Sie den Amazon-S3-Zielspeicherort aus, indem Sie auf S3 durchsuchen klicken.
- Aktualisierungsoptionen für Data Catalog – Erstellen einer Tabelle im Data Catalog und bei späteren Ausführungen das vorhandene Schema beibehalten und neue Partitionen hinzufügen muss ausgewählt sein um fortzufahren. Eine neue Iceberg-Tabelle schreiben mit AWS Glue benötigt die Data Catalog muss als Katalog für die Iceberg-Tabelle konfiguriert werden. Um eine bestehende Iceberg-Tabelle zu aktualisieren, die in der registriert wurde Data Catalog, wählen Data Catalog als Ziel.
  - Datenbank — Wählen Sie die Datenbank aus der Data Catalog.
  - Tabellenname – Geben Sie den Wert für Ihren Tabellennamen ein. Apache-Iceberg-Tabellennamen müssen ausschließlich in Kleinbuchstaben angegeben werden. Verwenden Sie bei Bedarf Unterstriche, da Leerzeichen nicht zulässig sind. Beispielsweise „data_lake_format_tables“.



Verwendung des Apache-Iceberg-Frameworks in Amazon-S3-Datenquellen

Verwendung des Apache-Iceberg-Frameworks in Data-Catalog-Datenquellen

1. Wählen Sie im Menü Quelle AWS Glue Studio Datenkatalog.
2. Wählen Sie auf in Registerkarte Datenquelleneigenschaften eine Datenbank und eine Tabelle aus.
3. AWS Glue Studio zeigt den Formattyp als Apache Iceberg und die Amazon S3 S3-URL an.



Verwendung des Apache-Iceberg-Frameworks in Amazon-S3-Datenquellen

Apache Iceberg ist nicht als Datenoption für Amazon S3 S3-Quellknoten verfügbar in AWS Glue Studio.

Herstellen von Verbindungen mit Datenquellen mithilfe von visuellen ETL-Aufträgen

Beim Erstellen eines neuen Jobs können Sie Verbindungen verwenden, um eine Verbindung zu Daten herzustellen, wenn Sie visuelle ETL-Jobs bearbeiten in AWS Glue. Sie können dies tun, indem

Sie Quellknoten hinzufügen, die Konnektoren zum Einlesen von Daten verwenden, und Zielknoten, um den Speicherort für das Ausschreiben von Daten anzugeben.

Themen

- Ändern der Eigenschaften eines Datenquellknotens
- Verwenden von Data-Catalog-Tabellen für die Datenquelle
- Verwenden eines Konnektors für die Datenquelle
- Verwenden von Dateien in Amazon S3 für die Datenquelle
- Verwenden einer Streaming-Datenquelle
- Referenzen

Ändern der Eigenschaften eines Datenquellknotens

Um die Datenquelleigenschaften anzugeben, wählen Sie zunächst einen Datenquellknoten im Auftragsdiagramm aus. Anschließend konfigurieren Sie rechts im Bereich mit den Knotendetails die Knoteneigenschaften.

Die Eigenschaften eines Datenquellknotens ändern

1. Rufen Sie im visuellen Editor einen neuen oder einen gespeicherten Auftrag auf.
2. Wählen Sie im Auftragsdiagramm einen Datenquellknoten aus.
3. Wählen Sie die Registerkarte Node properties (Knoteneigenschaften) im Bereich mit den Knotendetails aus und geben Sie die folgenden Informationen ein:
   - Name: (Optional) Geben Sie einen Namen ein, der dem Knoten im Auftragsdiagramm zugeordnet werden soll. Dieser Knotenname sollte in diesem Auftrag nur einmal vorkommen.
   - Node type (Knotentyp): Der Knotentyp bestimmt, welche Aktion der Knoten ausführt. Wählen Sie aus den Optionen bei Node type (Knotentyp) einen der Werte aus, die unter der Überschrift Data source (Datenquelle) stehen.
4. Passen Sie die Angaben unter Data source properties (Datenquelleneigenschaften) an. Weitere Informationen finden Sie in den folgenden Abschnitten:
   - Verwenden von Data-Catalog-Tabellen für die Datenquelle
   - Verwenden eines Konnektors für die Datenquelle
   - Verwenden von Dateien in Amazon S3 für die Datenquelle

- Verwenden einer Streaming-Datenquelle

5. (Optional) Nachdem Sie die Knoteneigenschaften und Datenquelleneigenschaften angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails das Schema für die Datenquelle sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.
6. (Optional) Nachdem Sie die Knoteneigenschaften und Datenquelleneigenschaften konfiguriert haben, sehen Sie auf der Registerkarte Data preview (Datenvorschau) im Bereich mit den Knotendetails eine Vorschau des Datensatzes aus Ihrer Datenquelle. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie dieses Feature verwenden, fallen Kosten an. Sobald Sie eine IAM-Rolle bereitstellen, wird dies berechnet.

Verwenden von Data-Catalog-Tabellen für die Datenquelle

Für alle Datenquellen außer Amazon S3 und Konnektoren muss eine Tabelle in der AWS Glue Data Catalog für den Quelltyp, den Sie wählen. AWS Glue erstellt die Datenkatalogtabelle nicht.

Einen Datenquellknoten basierend auf einer Data-Catalog-Tabelle konfigurieren

1. Rufen Sie im visuellen Editor einen neuen oder einen gespeicherten Auftrag auf.
2. Wählen Sie im Auftragsdiagramm einen Datenquellknoten aus.
3. Wählen Sie die Registerkarte Data source properties (Datenquelleneigenschaften) aus und geben Sie die folgenden Informationen ein:

   - S3-Quelltyp: (nur für Amazon S3 S3-Datenquellen) Wählen Sie die Option Wählen Sie eine Katalogtabelle aus, um eine vorhandene zu verwenden AWS Glue Data Catalog Tabelle.
   - Database (Datenbank): Wählen Sie die Datenbank im Data Catalog aus, die die Quelltabelle enthält, die Sie für diesen Auftrag verwenden möchten. Sie können über das Suchfeld eine Datenbank nach ihrem Namen suchen.
   - Table (Tabelle): Wählen Sie die Tabelle aus, die mit den Quelldaten verknüpft ist. Diese Tabelle muss bereits existieren in derAWS Glue Data Catalog. Sie können das Suchfeld verwenden, um anhand ihres Namens nach einer Tabelle zu suchen.

- Partition predicate (Partitionsprädikat): (Nur für Amazon-S3-Datenquellen) Geben Sie einen Booleschen Ausdruck ein, der auf Spark SQL nur mit Partitionierungsspalten basiert. Zum Beispiel: `"(year=='2020' and month=='04')"`
- Temporary directory (Temporäres Verzeichnis): (Nur für Amazon-Redshift-Datenquellen) Geben Sie einen Pfad für den Speicherort eines Arbeitsverzeichnisses in Amazon S3 ein, in das Ihr ETL-Auftrag temporäre Zwischenergebnisse schreiben kann.
- Role associated with the cluster (Mit dem Cluster verknüpfte Rolle): (Nur für Amazon-Redshift-Datenquellen) Geben Sie eine zu verwendende Rolle für Ihren ETL-Auftrag ein, die über Berechtigungen für Amazon Redshift -Cluster verfügt. Weitere Informationen finden Sie unter the section called “Datenquellen- und Datenzielberechtigungen”.

Verwenden eines Konnektors für die Datenquelle

Wenn Sie einen Konnektor für den Node type (Knotentyp) auswählen, befolgen Sie die Anweisungen im Abschnitt Erstellen von Aufträgen mit benutzerdefinierten Connectors, um die Konfiguration der Datenquelleneigenschaften abzuschließen.

Verwenden von Dateien in Amazon S3 für die Datenquelle

Wenn Sie Amazon S3 als Datenquelle auswählen, haben Sie die Wahl zwischen:

- einer Data-Catalog-Datenbank und -tabelle;
- einem Bucket, einem Ordner oder einer Datei in Amazon S3.

Wenn Sie einen Amazon S3 S3-Bucket als Datenquelle verwenden, AWS Glue erkennt das Schema der Daten am angegebenen Speicherort anhand einer der Dateien oder verwendet die von Ihnen angegebene Datei als Beispieldatei. Die Schemaerkennung wird aktiviert, wenn Sie die Schaltfläche Infer schema (Schema ableiten) auswählen. Wenn Sie den Amazon-S3-Speicherort oder die Beispieldatei ändern, müssen Sie erneut Infer schema (Schema ableiten) auswählen, um die Schemaerkennung mithilfe der neuen Informationen durchzuführen.

Einen Datenquellknoten konfigurieren, der direkt aus Dateien in Amazon S3 liest

1. Rufen Sie im visuellen Editor einen neuen oder einen gespeicherten Auftrag auf.
2. Wählen Sie im Auftragsdiagramm einen Datenquellknoten für eine Amazon-S3-Quelle aus.
3. Wählen Sie die Registerkarte Data source properties (Datenquelleneigenschaften) aus und geben Sie die folgenden Informationen ein:

- S3 source type (S3-Quelltyp): (Nur für Amazon-S3-Datenquellen) Wählen Sie die Option S3 location (S3-Speicherort) aus.
- S3 URL: Geben Sie den Pfad zur/zum Amazon-S3-Bucket, -Ordner oder -Datei ein, wo die Daten für Ihren Auftrag liegen. Sie können mit Browse S3 (S3 durchsuchen) den Pfad aus den Speicherorten auswählen, die für Ihr Konto verfügbar sind.
- Rekursiv: Wählen Sie diese Option, wenn Sie möchten AWS Glue um Daten aus Dateien in untergeordneten Ordnern am S3-Standort zu lesen.

  Wenn die untergeordneten Ordner partitionierte Daten enthalten, AWS Glue fügt dem Datenkatalog keine Partitionsinformationen hinzu, die in den Ordnernamen angegeben sind. Sehen Sie sich beispielsweise die folgenden Ordner in Amazon S3 an:

  ```
  S3://sales/year=2019/month=Jan/day=1
  S3://sales/year=2019/month=Jan/day=2
  ```

  Wenn Sie Rekursiv wählen und den `sales` Ordner als Ihren S3-Speicherort auswählen, dann AWS Glue liest die Daten in allen untergeordneten Ordnern, erstellt jedoch keine Partitionen für Jahr, Monat oder Tag.
- Data format (Datenformat): Wählen Sie das Format, in dem die Daten gespeichert werden. Sie können JSON, CSV oder Parquet wählen. Der von Ihnen gewählte Wert gibt an AWS Glue Job, wie man die Daten aus der Quelldatei liest.

  > **Note**
  >
  > Wenn Sie nicht das richtige Format für Ihre Daten wählen, AWS Glue könnte das Schema korrekt ableiten, aber der Job wird nicht in der Lage sein, die Daten aus der Quelldatei korrekt zu analysieren.

  Je nach gewähltem Format können Sie zusätzliche Konfigurationsoptionen eingeben.

  - JSON (JavaScript Objektnotation)
    - JsonPath: Geben Sie einen JSON-Pfad ein, der auf ein Objekt verweist, das zur Definition eines Tabellenschemas verwendet wird. JSON-Pfadausdrücke beziehen sich immer auf eine JSON-Struktur, genauso wie XPath Ausdrücke in Kombination mit einem XML-Dokument verwendet werden. Das „Root-Element-Objekt“ im JSON-Pfad ist stets mit $

gekennzeichnet, auch wenn es sich um ein Objekt oder ein Array handelt. Der JSON-Pfad kann in Punkt- oder Klammer-Notation angegeben werden.

Weitere Informationen zum JSON-Pfad finden Sie JsonPathauf der GitHub Website.

- Records in source files can span multiple lines (Akten in Quelldateien können sich über mehrere Zeilen erstrecken): Wählen Sie diese Option, wenn eine einzelne Akte sich über mehrere Zeilen in der CSV-Datei erstrecken kann.

- CSV (Comma Separated Values, durch Komma getrennte Werte)
  - Delimiter (Trennzeichen): Geben Sie das Trennzeichen für Spalteneinträge in der Zeile an, etwa `;` oder `,`.
  - Escape character (Escape-Zeichen): Geben Sie ein Zeichen ein, das als Escape-Zeichen verwendet werden soll. Zeichen, die unmittelbar auf dieses Escape-Zeichen folgen, werden nicht als Trennzeichen interpretiert.
  - Quote character (Zitatzeichen): Geben Sie das Zeichen ein, mit dem separate Zeichenfolgen in einem einzelnen Wert gruppiert werden sollen. Beispielsweise wählen Sie die Option Double quote (") (Doppeltes Anführungszeichen), wenn Sie Werte wie `"This is a single value"` in der CSV-Datei haben.
  - Records in source files can span multiple lines (Akten in Quelldateien können sich über mehrere Zeilen erstrecken): Wählen Sie diese Option, wenn eine einzelne Akte sich über mehrere Zeilen in der CSV-Datei erstrecken kann.
  - First line of source file contains column headers (Erste Zeile der Quelldatei enthält Spaltenüberschriften): Wählen Sie diese Option, wenn die erste Zeile in der CSV-Datei Spaltenüberschriften anstelle von Daten enthält.
- Parquet (Spaltenweise Speicherung von Apache Parquet)

  Für Daten im Parquet-Format gibt es keine zusätzlichen Einstellungen.
- Apache Hudi

  Für Daten, die im Apache Hudi-Format gespeichert sind, müssen keine zusätzlichen Einstellungen konfiguriert werden.
- Delta-See

  Für Daten, die im Delta Lake-Format gespeichert sind, müssen keine zusätzlichen Einstellungen konfiguriert werden.
- Excel

Für Daten, die im Excel-Format gespeichert sind, müssen keine zusätzlichen Einstellungen konfiguriert werden.

- Partition predicate (Partitionsprädikat): Um die Daten zu partitionieren, die aus der Datenquelle gelesen werden, geben Sie einen Booleschen Ausdruck ein, der auf Spark SQL nur mit Partitionierungsspalten basiert. Beispiel: `"(year=='2020' and month=='04')"`
- Erweiterte Optionen: Erweitern Sie diesen Abschnitt, wenn Sie möchten AWS Glue um das Schema Ihrer Daten anhand einer bestimmten Datei zu erkennen.
  - Schemainferenz: Wählen Sie die Option Wählen Sie eine Beispieldatei aus S3, wenn Sie eine bestimmte Datei verwenden möchten, anstatt sie zu lassen AWS Glue wählen Sie eine Datei. Schemainferenz ist für die Excel-Quelle nicht verfügbar.
  - Auto-sampled file (Automatische Beispieldatei): Geben Sie den Pfad zur Datei in Amazon S3 ein, die zum Ableiten des Schemas verwendet werden soll.

  Wenn Sie einen Datenquellknoten bearbeiten und die ausgewählte Beispieldatei ändern, wählen Sie die Option Reload Schema (Schema erneut laden) aus, um das Schema mithilfe der neuen Beispieldatei zu erkennen.

4. Wählen Sie die Schaltfläche Infer schema (Schema ableiten), um das Schema der Quelldateien in Amazon S3 zu erkennen. Wenn Sie den Amazon-S3-Speicherort oder die Beispieldatei ändern, müssen Sie erneut Infer schema (Schema ableiten) auswählen, um das Schema mithilfe der neuen Informationen abzuleiten.

Verwenden einer Streaming-Datenquelle

Sie können Streaming-Aufträge für Extract, Transform, Load (ETL) erstellen, die kontinuierlich ausgeführt werden und Daten aus Streaming-Quellen in Amazon Kinesis Data Streams, Apache Kafka und Amazon Managed Streaming for Apache Kafka (Amazon MSK) konsumieren.

Eigenschaften einer Streaming-Datenquelle konfigurieren

1. Rufen Sie im visuellen Diagrammeditor einen neuen oder einen gespeicherten Auftrag auf.
2. Wählen Sie im Diagramm einen Datenquellknoten für Kafka- oder Kinesis Data Streams aus.
3. Wählen Sie die Registerkarte Data source properties (Datenquelleneigenschaften) aus und geben Sie die folgenden Informationen ein:

Kinesis

- Kinesis-Quelltyp: Wählen Sie die Option Stream details, um den direkten Zugriff auf die Streaming-Quelle zu verwenden oder wählen Sie Data Catalog table (Tabelle des Data Catalog), um stattdessen die dort gespeicherten Informationen zu verwenden.

  Wenn Sie Stream details wählen, geben Sie die folgenden zusätzlichen Informationen an.

  - Standort des Datenstroms: Wählen Sie aus, ob der Strom dem aktuellen Benutzer zugeordnet ist oder ob er einem anderen Benutzer zugeordnet ist.
  - Region: Wählen Sie aus AWS-Region , wo der Stream existiert. Diese Informationen werden verwendet, um den ARN für den Zugriff auf den Datenstrom zu erstellen.
  - Stream ARN: Geben Sie den Amazon-Ressourcenname (ARN) für den Kinesis-Datenstrom ein. Wenn sich der Stream im Girokonto befindet, können Sie den Stream-Namen aus der Dropdown-Liste auswählen. Sie können über das Suchfeld einen Datenbank-Stream nach ihrem Namen oder ARN suchen.
  - Datenformat: Wählen Sie das vom Datenstrom verwendete Format aus der Liste aus.

    AWS Glue erkennt das Schema automatisch anhand der Streaming-Daten.

  Wenn Sie Data Catalog table (Data Catalogtabelle) wählen, geben Sie die folgenden zusätzlichen Informationen an.

  - Datenbank: (Optional) Wählen Sie die Datenbank im AWS Glue Datenkatalog, der die Tabelle enthält, die Ihrer Streaming-Datenquelle zugeordnet ist. Sie können über das Suchfeld eine Datenbank nach ihrem Namen suchen.
  - Table (Tabelle): (Optional) Wählen Sie die Tabelle aus, die mit den Quelldaten verknüpft ist. Diese Tabelle muss bereits in der AWS Glue Datenkatalog. Sie können über das Suchfeld eine Tabelle nach ihrem Namen suchen.
  - Schema erkennen: Wählen Sie diese Option, um AWS Glue erkennt das Schema anhand der Streaming-Daten, anstatt die Schemainformationen in einer Datenkatalogtabelle zu verwenden. Diese Option wird automatisch aktiviert, wenn Sie die Option Stream details ausgewählt haben.
- Startposition: Standardmäßig verwendet der ETL-Auftrag die Option Earliest (frühestens), was bedeutet, dass sie Daten liest, beginnend mit dem ältesten verfügbaren Datensatz im Stream. Sie können stattdessen Latest (Neuste), was darauf hinweist, dass der ETL-Auftrag kurz nach dem letzten Datensatz im Stream zu lesen beginnen sollte.

- Window size (Fenstergröße): Standardmäßig verarbeitet und schreibt der ETL-Auftrag Daten in 100-Sekunden-Fenstern. Dadurch können Daten effizient verarbeitet und Aggregationen für Daten ausgeführt werden, die später als erwartet eintreffen. Sie können die Größe dieses Zeitfensters ändern, um die Aktualität oder Aggregationsgenauigkeit zu erhöhen.

  AWS Glue Streaming-Jobs verwenden Checkpoints statt Job-Lesezeichen, um die gelesenen Daten nachzuverfolgen.
- Connection options (Verbindungsoptionen): Klappen Sie diesen Abschnitt aus, um Schlüssel-Wert-Paare hinzuzufügen und so zusätzliche Verbindungsoptionen anzugeben. Weitere Informationen zu den Optionen, die Sie hier angeben können, finden Sie unter "connectionType": "kinesis" im AWS Glue -Benutzerhandbuch.

Kafka

- Apache-Kafka-Quelle: Wählen Sie die Option Stream details, um den direkten Zugriff auf die Streaming-Quelle zu verwenden oder wählen Sie Data Catalog table (Tabelle des Data Catalog), um stattdessen die dort gespeicherten Informationen zu verwenden.

  Wenn Sie Data Catalog table (Data Catalogtabelle) wählen, geben Sie die folgenden zusätzlichen Informationen an.
  - Datenbank: (Optional) Wählen Sie die Datenbank im AWS Glue Datenkatalog, der die Tabelle enthält, die Ihrer Streaming-Datenquelle zugeordnet ist. Sie können über das Suchfeld eine Datenbank nach ihrem Namen suchen.
  - Table (Tabelle): (Optional) Wählen Sie die Tabelle aus, die mit den Quelldaten verknüpft ist. Diese Tabelle muss bereits in der AWS Glue Datenkatalog. Sie können über das Suchfeld eine Tabelle nach ihrem Namen suchen.
  - Schema erkennen: Wählen Sie diese Option, um AWS Glue erkennt das Schema anhand der Streaming-Daten, anstatt die Schemainformationen in einer Datenkatalogtabelle zu speichern. Diese Option wird automatisch aktiviert, wenn Sie die Option Stream details ausgewählt haben.

  Wenn Sie Stream details wählen, geben Sie die folgenden zusätzlichen Informationen an.
  - Verbindungsname: Wählen Sie AWS Glue Verbindung, die die Zugriffs- und Authentifizierungsinformationen für den Kafka-Datenstrom enthält. Sie müssen eine Verbindung mit Kafka-Streaming-Datenquellen verwenden. Wenn keine Verbindung

besteht, können Sie die AWS Glue Konsole, um eine Verbindung für Ihren Kafka-Datenstrom herzustellen.

- Topic-Name: Geben Sie den Namen des Themas ein, aus dem gelesen werden soll.
- Datenformat: Wählen Sie das Format aus, das beim Lesen von Daten aus dem Kafka-Ereignisstream verwendet werden soll.

- Startposition: Standardmäßig verwendet der ETL-Auftrag die Option Earliest (frühestens), was bedeutet, dass sie Daten liest, beginnend mit dem ältesten verfügbaren Datensatz im Stream. Sie können stattdessen Latest (Neuste), was darauf hinweist, dass der ETL-Auftrag kurz nach dem letzten Datensatz im Stream zu lesen beginnen sollte.
- Window size (Fenstergröße): Standardmäßig verarbeitet und schreibt der ETL-Auftrag Daten in 100-Sekunden-Fenstern. Dadurch können Daten effizient verarbeitet und Aggregationen für Daten ausgeführt werden, die später als erwartet eintreffen. Sie können die Größe dieses Zeitfensters ändern, um die Aktualität oder Aggregationsgenauigkeit zu erhöhen.

  AWS Glue Streaming-Jobs verwenden Checkpoints anstelle von Job-Lesezeichen, um die gelesenen Daten zu verfolgen.
- Connection options (Verbindungsoptionen): Klappen Sie diesen Abschnitt aus, um Schlüssel-Wert-Paare hinzuzufügen und so zusätzliche Verbindungsoptionen anzugeben. Weitere Informationen zu den Optionen, die Sie hier angeben können, finden Sie unter "connectionType": "kafka" im AWS Glue -Benutzerhandbuch.

> **Note**
>
> Aktuell sind Datenvorschauen bei Streaming-Datenquellen nicht möglich.

Referenzen

Bewährte Methoden

- Erstellen Sie eine ETL-Servicepipeline, um Daten inkrementell von Amazon S3 zu Amazon Redshift laden AWS Glue

ETL-Programmierung

- Verbindungstypen und Optionen für ETL in AWS Glue
- JDBC-connectionType-Werte
- Erweiterte Optionen für das Verschieben von und zu Daten Amazon Redshift

Konfigurieren von Datenzielknoten

Das Datenziel ist der Ort, wohin der Auftrag die transformierten Daten schreibt.

Übersicht über mögliche Datenziele

Das Datenziel (auch data sink (Datensenke)) kann Folgendes sein:

- S3 – der Auftrag schreibt die Daten in eine Datei im angegebenen Format in den ausgewählten Amazon-S3-Speicherort.

  Wenn Sie Partitionsspalten für das Datenziel konfigurieren, schreibt der Auftrag den Datensatz in Amazon-S3-Verezichnisse basierend auf dem Partitionsschlüssel .
- AWS Glue Data Catalog – der Auftrag verwendet die Informationen, die der Tabelle im Data Catalog zugeordnet sind, um die Ausgabedaten an einen Zielspeicherort zu schreiben.

  Sie können die Tabelle manuell oder mit dem Crawler erstellen. Sie können Tabellen im Data Catalog auch mit AWS CloudFormation -Vorlagen erstellen.
- Ein Konnektor — Ein Konnektor ist ein Code, der die Kommunikation zwischen Ihrem Datenspeicher und AWS Glue. Der Job verwendet den Konnektor und die zugehörige Verbindung, um die Ausgabedaten an einen Zielort zu schreiben. Sie können entweder einen Connector abonnieren, der unter angeboten wird AWS Marketplace, oder Sie können Ihren eigenen benutzerdefinierten Connector erstellen. Weitere Informationen finden Sie unter Konnektoren werden hinzugefügt zu AWS Glue Studio.

Sie können den Data Catalog aktualisieren lassen, wenn Ihr Auftrag in ein Amazon-S3-Datenziel schreibt. Dadurch ist es einfacher, die Tabellen auf dem neuesten Stand zu halten, wenn sich das Schema oder die Partitionen ändern (im Vergleich zur Aktualisierung per Crawler). Diese Methode vereinfacht die Bereitstellung Ihrer Daten für Analysen, da dadurch optional neue Tabellen zum Data Catalog hinzugefügt, Tabellenpartitionen geändert und das Schema Ihrer Tabellen direkt aus dem Auftrag aktualisiert werden.

Bearbeiten des Datenzielknotens

Das Datenziel ist der Ort, wohin der Auftrag die transformierten Daten schreibt.

Einen Datenzielknoten im Auftragsdiagramm hinzufügen oder konfigurieren

1. (Optional) Wenn Sie einen Zielknoten hinzufügen müssen, wählen Sie Target (Ziel) in der Symbolleiste oben im visuellen Editor aus und dann entweder S3 oder Glue Data Catalog aus.

   - Wenn Sie S3 als Ziel festlegen, schreibt der Auftrag den Datensatz in eine oder mehrere Dateien im angegebenen Amazon-S3-Speicherort.
   - Wenn Sie AWS Glue Data Catalog als Ziel festlegen, schreibt der Auftrag in einen Speicherort, der von der im Data Catalog ausgewählten Tabelle beschrieben wird.
2. Wählen Sie im Auftragsdiagramm einen Datenzielknoten aus. Wenn Sie einen Knoten auswählen, erscheint rechts das Fenster mit den Knotendetails.
3. Wählen Sie die Registerkarte Node properties (Knoteneigenschaften) aus und geben Sie die folgenden Informationen ein:

   - Name: Geben Sie einen Namen ein, der dem Knoten im Auftragsdiagramm zugeordnet werden soll.
   - Node type (Knotentyp): Es sollte bereits ein Wert ausgewählt sein, Sie können ihn aber auch nach Bedarf ändern.
   - Node parents (Übergeordnete Knoten): Der übergeordnete Knoten ist der Knoten im Auftragsdiagramm, der die Ausgabedaten bereitstellt, die Sie in den Zielspeicherort schreiben möchten. Für ein vorausgefülltes Auftragsdiagramm sollte beim Zielknoten bereits der übergeordnete Knoten ausgewählt sein. Wenn kein übergeordneter Knoten angezeigt wird, wählen Sie einen übergeordneten Knoten aus der Liste aus.

     Ein Zielknoten hat einen einzelnen übergeordneten Knoten.
4. Passen Sie die Angaben unter Data target properties (Datenzieleigenschaften) an. Weitere Informationen finden Sie in den folgenden Abschnitten:

   - Verwenden von Amazon S3 als Datenziel
   - Verwenden von Data-Catalog-Tabellen für das Datenziel
   - Verwenden eines Konnektoren für das Datenziel
5. (Optional) Nachdem Sie die Eigenschaften des Datenzielknotens angepasst haben, können Sie mit der Registerkarte Output schema (Ausgabeschema) im Bereich mit den Knotendetails

das Ausgabeschema für die Daten sehen. Wenn Sie diese Registerkarte zum ersten Mal für einen Knoten in Ihrem Auftrag auswählen, werden Sie aufgefordert, eine IAM-Rolle für den Zugriff auf die Daten anzugeben. Wenn Sie keine IAM-Rolle auf der Registerkarte Job details (Auftragsdetails) angegeben haben, werden Sie aufgefordert, hier eine IAM-Rolle einzugeben.

Verwenden von Amazon S3 als Datenziel

Für alle Datenquellen außer Amazon S3 und Konnektoren muss eine Tabelle in der AWS Glue Data Catalog für den Quelltyp, den Sie wählen. AWS Glue Studio erstellt die Datenkatalogtabelle nicht.

Einen Datenzielknoten konfigurieren, der in Amazon S3 schreibt

1. Rufen Sie im visuellen Editor einen neuen oder einen gespeicherten Auftrag auf.
2. Wählen Sie im Auftragsdiagramm einen Datenquellknoten aus.
3. Wählen Sie die Registerkarte Data source properties (Datenquelleneigenschaften) aus und geben Sie die folgenden Informationen ein:

   - Format: Wählen Sie ein Format aus der Liste aus. Die verfügbaren Formattypen für die Datenergebnisse sind:
     - JSON: JavaScript Objektnotation.
     - CSV: Comma Separated Values (durch Komma getrennte Werte)
     - Avro: Apache Avro-JSON-Binärdaten
     - Parquet: Ein benutzerdefinierter Parquet-Writer-Typ, der für das Datenformat `DynamicFrames` AS optimiert ist. Anstatt ein vorberechnetes Schema für die Daten zu benötigen, berechnet und modifiziert er das Schema dynamisch.
     - ORC: Das Apache-Format „Optimized Row Columnar“
     - Apache Hudi: Ein Open-Source-Data-Lake-Speicher-Framework, das die inkrementelle Datenverarbeitung und die Entwicklung von Datenpipelines vereinfacht.
     - Apache Iceberg: Ein leistungsstarkes Tabellenformat, das genau wie eine SQL-Tabelle funktioniert.
     - Delta Lake: Ein Open-Source-Data-Lake-Speicher-Framework, mit dem Sie ACID-Transaktionen durchführen, die Metadatenverarbeitung skalieren und Streaming und Batch-Datenverarbeitung vereinheitlichen können.
     - XML: Extensible Markup Language (XML).
     - Tableau Hyper: Die speicherinterne Daten-Engine-Technologie von Tableau.

Weitere Informationen zu diesen Formatoptionen finden Sie unter Formatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue im Entwicklerhandbuch für AWS Glue .

- Komprimierungstyp: Sie können wählen, ob Sie die Daten optional mit den Dateitypen `CSVJSON`, oder komprimieren möchten`Parquet`. Die Standardeinstellung ist keine Komprimierung bzw. None (Keine).

Dateityp	Komprimierungen
JSON/CSV/XML	GZIP, BROTLI BZIP2, DEFLATE, Snappy LZ4
Parquet	Schnell, LAO, BROTLI, GZIP LZ4
ORC	Snappy, ZLIB, Unkomprimiert, LZO
Avro	GZIP, BROTLI, BZIP2 DEFLATE, Snappy LZ4
Delta Lake	GZIP, BROTLI, DEFLATE, Snappy BZIP2 LZ4
Apache Hudi	GZIP, LAO, Snappy
Apache Iceberg	GZIP, LZO, Snappy
Hyper-Tabell	Keine

- S3 Target Location (S3-Zielspeicherort): Der Amazon-S3-Bucket und -Speicherort für die Datenausgabe. Wählen Sie Browse S3 (S3 durchsuchen), um die Amazon-S3-Buckets zu sehen, auf die Sie Zugriff haben. Wählen Sie einen davon als Zielspeicherort aus.
- Aktualisierungsoptionen für den Data Catalog
  - Do not update the Data Catalog (Data Catalog nicht aktualisieren): (Standard) Wählen Sie diese Option, wenn der Auftrag den Data Catalog nicht aktualisieren soll, selbst wenn sich das Schema ändert oder neue Partitionen hinzukommen.
  - Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions (Eine Tabelle im Data Catalog erstellen und bei folgenden Ausführungen das Schema aktualisieren und neue Partitionen hinzufügen): Wenn Sie diese Option wählen,

erstellt der Auftrag die Tabelle im Data Catalog bei der ersten Ausführung des Auftrags. Bei nachfolgenden Ausführungen aktualisiert der Auftrag die Data-Catalog-Tabelle, wenn sich das Schema ändert oder neue Partitionen hinzugefügt werden.

Sie müssen auch eine Datenbank aus dem Data Catalog auswählen und einen Tabellennamen eingeben.

- Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions (Eine Tabelle im Data Catalog erstellen und bei folgenden Ausführungen das bestehende Schema beibehalten und neue Partitionen hinzufügen): Wenn Sie diese Option wählen, erstellt der Auftrag die Tabelle im Data Catalog bei der ersten Ausführung des Auftrags. Bei nachfolgenden Ausführungen fügt der Auftrag lediglich neue Partitionen in die Data-Catalog-Tabelle ein.

  Sie müssen auch eine Datenbank aus dem Data Catalog auswählen und einen Tabellennamen eingeben.

- Dateipartitionierung: Wählen Sie aus, in welcher Art von Partitionierung Sie die Ausgabe speichern möchten.
  - Dateien automatisch generieren (empfohlen): Dies ist der Standardwert für die Anzahl der generierten Dateien.
  - Ausgabe mehrerer Dateien: Geben Sie die Anzahl der gewünschten Dateiausgaben an. Verwenden Sie für eine optimale Leistung den Standardwert für die automatisch generierte Anzahl von Dateien.
- Partition keys (Partitionsschlüssel): Wählen Sie aus, welche Spalten als Partitionierungsschlüssel in der Ausgabe verwendet werden sollen. Um weitere Partitionsschlüssel hinzuzufügen, wählen Sie Add a partition key (Partitionsschlüssel hinzufügen) aus.

Die Dateipartitionierung wird für Tableau Hyper als Zielformat nicht unterstützt.

Verwenden von Data-Catalog-Tabellen für das Datenziel

Für alle Datenquellen außer Amazon S3 und Konnektoren muss eine Tabelle in der AWS Glue Data Catalog für den Zieltyp, den Sie wählen. AWS Glue Studio erstellt die Datenkatalogtabelle nicht.

Die Dateneigenschaften für ein Ziel konfigurieren, das eine Data-Catalog-Tabelle verwendet

1. Rufen Sie im visuellen Editor einen neuen oder einen gespeicherten Auftrag auf.

2. Wählen Sie im Auftragsdiagramm einen Datenzielknoten aus.
3. Wählen Sie die Registerkarte Data target properties (Datenzieleigenschaften) aus und geben Sie die folgenden Informationen ein:

   - Database (Datenbank): Wählen Sie in der Liste die Datenbank aus, die die Tabelle enthält, die Sie als Ziel verwenden möchten. Diese Datenbank muss bereits im Data Catalog vorhanden sein.
   - Table (Tabelle): Wählen Sie die Tabelle aus, die das Schema Ihrer Ausgabedaten definiert. Diese Tabelle muss bereits im Data Catalog vorhanden sein.

     Eine Tabelle im Data Catalog besteht aus den Namen der Spalten, Datentypdefinitionen, Partitionsinformationen und anderen Metadaten zu einem Zieldatensatz. Der Auftrag schreibt in einen Speicherort, der in dieser Tabelle im Data Catalog beschrieben wird.

     Weitere Informationen zum Erstellen von Tabellen im Data Catalog finden Sie unter Definieren von Tabellen im Data Catalog im AWS Glue -Entwicklerhandbuch.
   - Aktualisierungsoptionen für den Data Catalog
     - Do not change table definition (Tabellendefinition nicht ändern): (Standard) Wählen Sie diese Option, wenn der Auftrag den Data Catalog nicht aktualisieren soll, selbst wenn sich das Schema ändert oder neue Partitionen hinzukommen.
     - Update schema and add new partitions (Schema aktualisieren und neue Partitionen hinzufügen): Wenn Sie diese Option wählen, aktualisiert der Auftrag die Data-Catalog-Tabelle, wenn sich das Schema ändert oder neue Partitionen hinzugefügt werden.
     - Keep existing schema and add new partitions (Bestehendes Schema beibehalten und neue Partitionen hinzufügen): Wenn Sie diese Option wählen, fügt der Auftrag lediglich neue Partitionen zur Data-Catalog-Tabelle hinzu.
     - Partition keys (Partitionsschlüssel): Wählen Sie aus, welche Spalten als Partitionierungsschlüssel in der Ausgabe verwendet werden sollen. Um weitere Partitionsschlüssel hinzuzufügen, wählen Sie Add a partition key (Partitionsschlüssel hinzufügen) aus.

Verwenden eines Konnektoren für das Datenziel

Wenn Sie einen Konnektor für den Node type (Knotentyp) auswählen, folgen Sie den Anweisungen unter Erstellen von Aufträgen mit benutzerdefinierten Connectors, um die Konfiguration der Datenzieleigenschaften abzuschließen.

Bearbeiten oder Hochladen eines Auftragsskripts

Verwenden Sie AWS Glue Studio Visueller Editor, um das Job-Skript zu bearbeiten oder Ihr eigenes Skript hochzuladen.

Sie können den visuellen Editor nur dann zum Bearbeiten von Job-Knoten verwenden, wenn die Jobs mit erstellt wurden AWS Glue Studio. Wenn der Job mit dem erstellt wurde AWS Glue Konsole, über API-Befehle oder über die Befehlszeilenschnittstelle (CLI) können Sie den Skripteditor verwenden in AWS Glue Studio um das Job-Skript, die Parameter und den Zeitplan zu bearbeiten. Sie können auch das Skript für einen Job bearbeiten, der in erstellt wurde AWS Glue Studio indem Sie den Job in den reinen Skriptmodus konvertieren.

Das Auftragsskript bearbeiten oder ein eigenes Skript hochladen

1. Wenn Sie einen neuen Auftrag erstellen, wählen Sie auf der Registerkarte Jobs (Aufträge) die Option Spark script editor (Spark-Skripteditor) aus, um einen Spark-Auftrag zu erstellen. Alternativ erstellen Sie mit dem Python Shell script editor (Python-Shell-Skripteditor) einen Python-Shell-Auftrag. Sie können entweder ein neues Skript schreiben oder ein vorhandenes Skript hochladen. Wenn Sie Spark script editor (Spark-Skripteditor) auswählen, können Sie entweder ein Scala- oder Python-Skript schreiben bzw. hochladen. Wenn Sie Python Shell script editor (Python-Shell-Skripteditor) auswählen, können Sie nur ein Python-Skript schreiben oder hochladen.

   Nachdem Sie die Option zum Erstellen eines neuen Auftrags ausgewählt haben, erscheint der Bereich Options (Optionen). Hier können Sie entweder mit einem Starterskript loslegen (Create a new script with boilerplate code (Neues Skript mit Standardcode erstellen)) oder eine lokale Datei hochladen, die als Auftragsskript verwendet werden soll.

   Wenn Sie Spark script editor (Spark-Skripteditor) auswählen, können Sie Scala- oder Python-Skriptdateien hochladen. Scala-Skripts müssen die Dateiendung `.scala` haben. Python-Skripts müssen als Dateien vom Typ Python zu erkennen sein. Wenn Sie Python Shell script editor (Python-Shell-Skripteditor) auswählen, können Sie nur Python-Skriptdateien hochladen.

   Wenn Sie Ihre Auswahl getroffen haben, wählen Sie Create (Erstellen) aus, um den Auftrag zu erstellen und den visuellen Editor zu öffnen.
2. Wechseln Sie zum visuellen Auftragseditor für den neuen bzw. gespeicherten Auftrag und wählen Sie dann die Registerkarte Script (Skript) aus.

3. Wenn Sie keinen neuen Auftrag mit einer der Optionen des Skript-Editors erstellt haben und das Skript für einen vorhandenen Auftrag noch nie bearbeitet haben, zeigt die Registerkarte Script (Skript) die Überschrift Script (Locked) (Skript (gesperrt)). Das bedeutet, dass der Skripteditor im schreibgeschützten Modus ist. Wählen Sie Edit script (Skript bearbeiten), um das Skript für die Bearbeitung zu entsperren.

   Um das Skript editierbar zu machen, AWS Glue Studio konvertiert Ihren Job von einem visuellen Job in einen reinen Skriptauftrag. Wenn Sie das Skript für die Bearbeitung entsperren, können Sie den visuellen Editor für diesen Auftrag nach dem Speichern nicht mehr verwenden.

   Wählen Sie im Bestätigungsfenster Confirm (Bestätigen) aus, um fortzufahren, oder Cancel (Abbrechen), um den Auftrag für die visuelle Bearbeitung verfügbar zu halten.

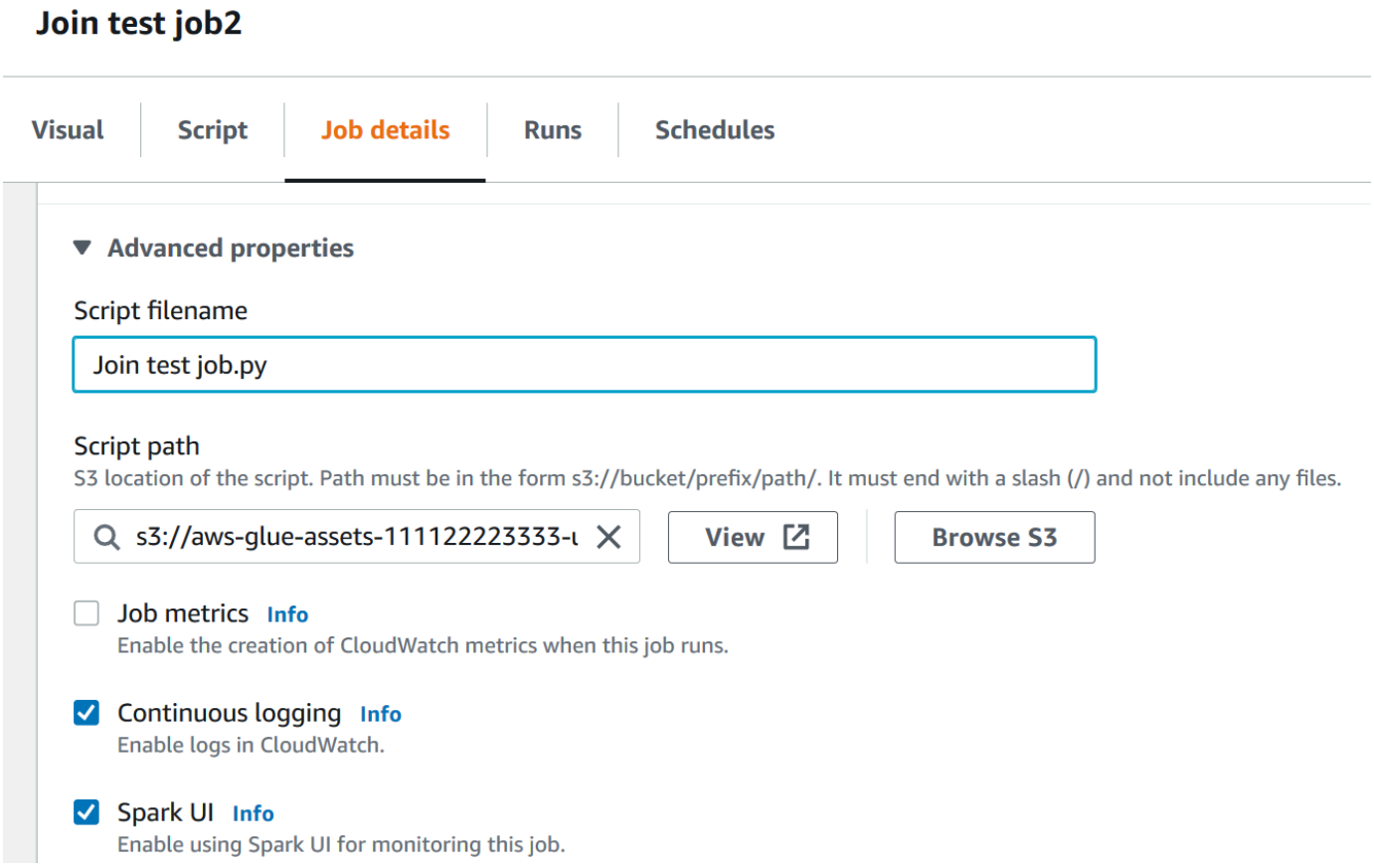
   Wenn Sie Confirm (Bestätigen) auswählen, verschwindet die Registerkarte Visual (Visuell) aus dem Editor. Sie können Folgendes verwenden … AWS Glue Studio um das Skript mit dem Skripteditor zu ändern, die Jobdetails oder den Zeitplan zu ändern oder Jobausführungen anzuzeigen.

   > **Note**
   >
   > Bis Sie den Auftrag speichern, ist die Konvertierung in einen reinen Skriptauftrag nicht dauerhaft. Wenn Sie die Konsolenwebseite aktualisieren oder den Auftrag vor dem Speichern schließen und im visuellen Editor erneut öffnen, können Sie die einzelnen Knoten weiterhin im visuellen Editor bearbeiten.

4. Bearbeiten Sie das Skript nach Bedarf.

   Wenn Sie mit der Bearbeitung des Skripts fertig sind, wählen Sie Save (Speichern) aus, um den Auftrag zu speichern und den Auftrag dauerhaft vom visuellen Modus zum den reinen Skriptmodus zu konvertieren.

5. (Optional) Sie können das Skript von der AWS Glue Studio Konsole, indem Sie auf der Registerkarte Skript auf die Schaltfläche Herunterladen klicken. Wenn Sie diese Schaltfläche auswählen, wird ein neues Browserfenster geöffnet, in dem das Skript von seinem Speicherort in Amazon S3 angezeigt wird. Die Parameter Script filename (Dateiname des Skripts) und Script path (Skript-Pfad) auf der Registerkarte Job details (Auftragsdetails) des Auftrags bestimmen den Namen und den Speicherort der Skriptdatei in Amazon S3.



Wenn Sie den Job speichern, AWS Glue speichern Sie das Jobskript an dem in diesen Feldern angegebenen Ort. Wenn Sie die Skriptdatei an diesem Speicherort in Amazon S3 ändern, AWS Glue Studio lädt das geänderte Skript, wenn Sie den Job das nächste Mal bearbeiten.

Erstellen und Bearbeiten von Scala-Skripten in AWS Glue Studio

Wenn Sie den Skripteditor zum Erstellen eines Auftrags auswählen, ist die Programmiersprache für den Auftrag standardmäßig `Python 3`. Wenn Sie sich dafür entscheiden, ein neues Skript zu schreiben, anstatt ein Skript hochzuladen, AWS Glue Studio startet ein neues Skript mit in Python geschriebenem Boilerplate-Text. Wenn Sie stattdessen ein Scala-Skript schreiben möchten, müssen Sie zunächst den Skripteditor so konfigurieren, dass er Scala verwendet.

> **Note**
>
> Wenn Sie Scala als Programmiersprache für den Auftrag wählen und den Auftrag mit dem visuellen Editor entwerfen, wird das generierte Auftragsskript in Scala geschrieben. Dabei sind keine weiteren Aktionen erforderlich.

Um ein neues Scala-Skript zu schreiben AWS Glue Studio

1. Erstellen Sie einen neuen Auftrag mit der Option Spark script editor (Spark-Skripteditor).
2. Wählen Sie unter Optionen die Option Create a new script with boilerplate code (Neues Skript mit Standardcode erstellen) aus.
3. Wählen Sie die Registerkarte Job details (Auftragsdetails) aus und legen Sie die Language (Sprache) auf `Scala` fest (anstelle von `Python 3`).

   > **Note**
   >
   > Die Eigenschaft Type (Typ) des Auftrags ist automatisch `Spark`, wenn Sie einen Auftrag mit der Option Spark script editor (Spark-Skripteditor) erstellen.

4. Wählen Sie die Registerkarte Script (Skript) aus.
5. Löschen Sie den Python-Standardtext. Sie können ihn durch den folgenden Scala-Standardtext ersetzen.

```
import com.amazonaws.services.glue.{DynamicRecord, GlueContext}
import org.apache.spark.SparkContext
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job

object MyScript {
  def main(args: Array[String]): Unit = {
    val sc: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(sc)

    }
}
```

6. Schreiben Sie Ihr Scala-Auftragsskript in den Editor. Fügen Sie nach Bedarf weitere `import`-Anweisungen hinzu.

Erstellen und Bearbeiten von Python-Shell-Jobs in AWS Glue Studio

Wenn Sie den Python-Shell-Skripteditor zum Erstellen eines Auftrags auswählen, können Sie ein vorhandenes Python-Skript hochladen oder ein neues schreiben. Wenn Sie ein neues Skript schreiben möchten, wird dem neuen Python-Auftragsskript Standardcode hinzugefügt.

Einen neuen Python-Shell-Auftrag erstellen

Anweisungen dazu finden Sie unter Jobs starten in AWS Glue Studio.

Die Auftragseigenschaften, die für Python-Shell-Aufträge unterstützt werden, sind nicht identisch mit denen, die für Spark-Aufträge unterstützt werden. In der folgenden Liste sehen Sie die Änderungen an den verfügbaren Auftragsparametern für Python-Shell-Aufträge auf der Registerkarte Job details (Auftragsdetails).

- Die Auftragseigenschaft Type (Typ) ist automatisch `Python Shell` und kann nicht geändert werden.
- Anstelle von Language (Sprache) hat der Auftrag die Eigenschaft Python version (Python-Version). Derzeit wurden Python-Shell-Jobs erstellt in AWS Glue Studio benutze Python 3.6.
- Die Eigenschaft Glue version (Glue-Version) ist nicht verfügbar, da sie nicht auf Python-Shell-Aufträge angewendet wird.
- Anstelle von Worker type (Worker-Typ) und Number of workers (Anzahl der Worker) erscheint die Eigenschaft Data processing units (Datenverarbeitungseinheiten). Diese Jobeigenschaft bestimmt, wie viele Datenverarbeitungseinheiten (DPUs) von der Python-Shell bei der Ausführung des Jobs verbraucht werden.
- Die Eigenschaft Job bookmark (Auftragslesezeichen) ist nicht verfügbar, da sie für Python-Shell-Aufträge nicht unterstützt wird.
- Unter Advanced properties (Erweiterte Eigenschaften) sind die folgenden Eigenschaften bei Python-Shell-Aufträgen nicht verfügbar.
  - Auftragsmetriken
  - Kontinuierliche Protokollierung
  - Spark UI und Spark UI logs path (Pfad der Spark-UI-Protokolle)
  - Dependant jars path (Abhängiger Jars-Pfad) unter der Überschrift Libraries (Bibliotheken)

Ändern der übergeordneten Knoten für einen Knoten im Auftragsdiagramm

Sie können die übergeordneten Knoten eines Knotens ändern, um Knoten innerhalb des Auftragsdiagramms zu verschieben oder eine Datenquelle für einen Knoten zu ändern.

Den übergeordneten Knoten ändern

1. Wählen Sie im Auftragsdiagramm den Knoten aus, den Sie ändern möchten.

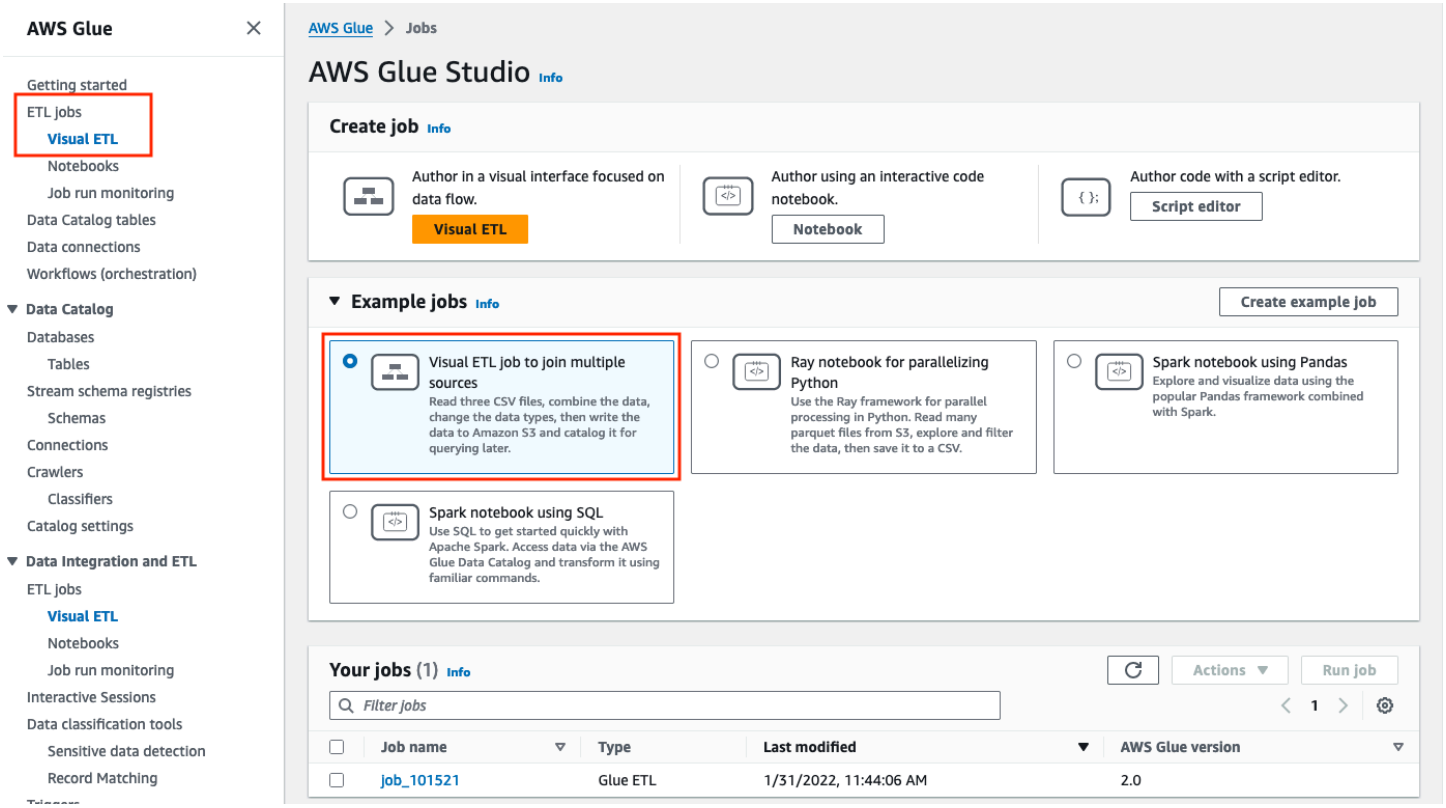
2. Entfernen Sie das aktuelle übergeordnete Element für den Knoten im Bereich mit den Knotendetails (Registerkarte Node properties (Knoteneigenschaften), Überschrift Node parents (Übergeordnete Knoten).
3. Wählen Sie einen neuen übergeordneten Knoten aus der Liste aus.
4. Ändern Sie die anderen Eigenschaften des Knotens nach Bedarf, damit sie dem neu ausgewählten übergeordneten Knoten entsprechen.

Wenn Sie versehentlich einen Knoten geändert haben, können Sie das mit der Schaltfläche Undo (Rückgängig) auf der Symbolleiste ungeschehen machen.

Löschen von Knoten aus dem Auftragsdiagramm

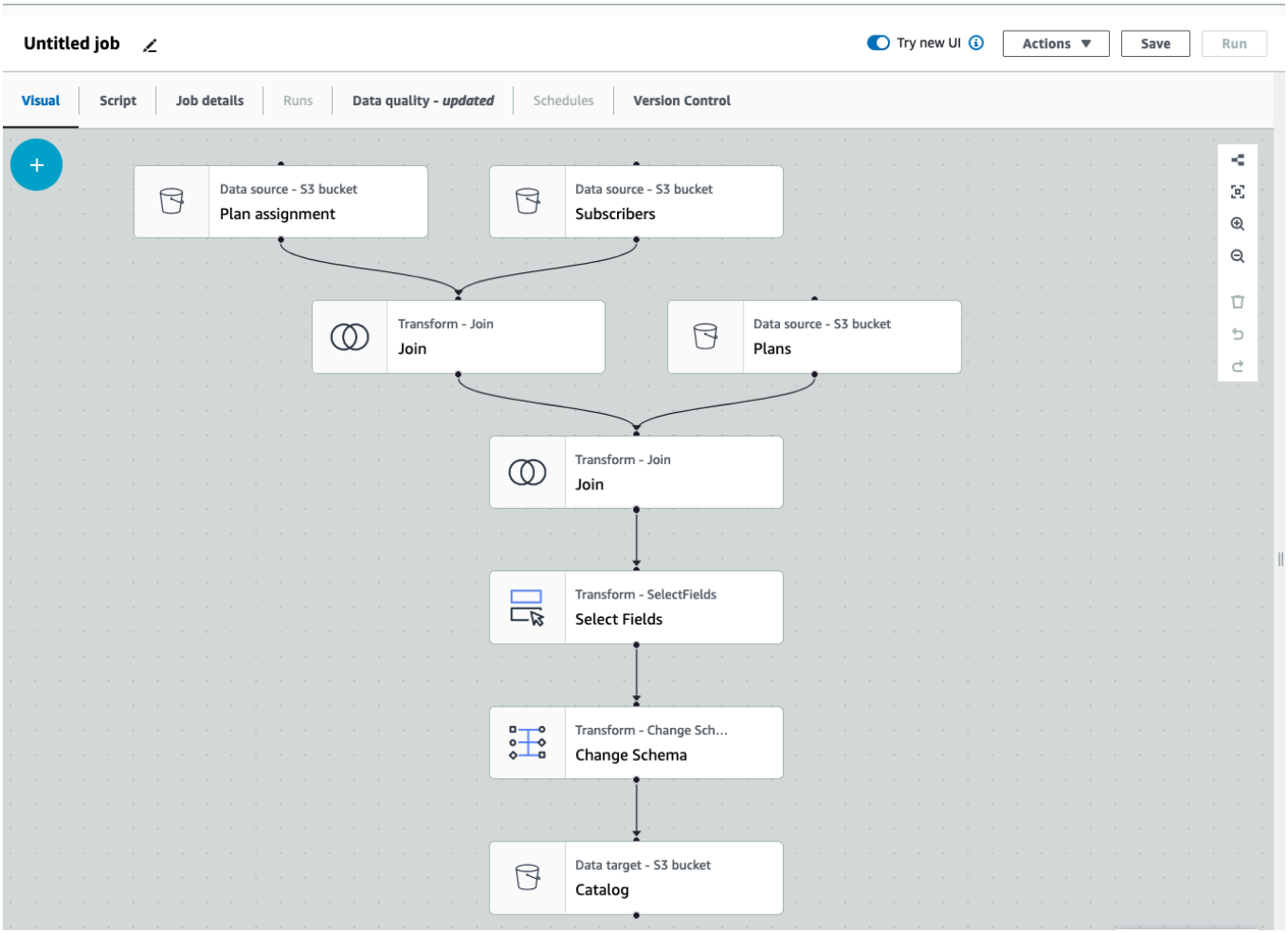
Bei der Arbeit mit Visual ETL-Jobs können Sie Knoten aus der Arbeitsfläche entfernen, ohne Knoten, die mit dem entfernten Knoten verbunden sind, erneut hinzufügen oder neu strukturieren zu müssen.

Im Beispiel unten können Sie dem Beispiel folgen, indem Sie ETL-Jobs > Visual ETL und dann unter Beispielaufträge Visual ETL-Job auswählen, um mehrere Quellen zu verbinden. Wählen Sie Beispieljob erstellen aus, um einen Job zu erstellen, und folgen Sie den nachfolgenden Schritten.



Um einen Knoten aus der Arbeitsfläche zu entfernen

1. Wählen Sie in der AWS Glue Konsole im Navigationsmenü Visual ETL und wählen Sie einen vorhandenen Job aus. Auf der Job-Canvas wird der Beispieljob wie unten dargestellt angezeigt.

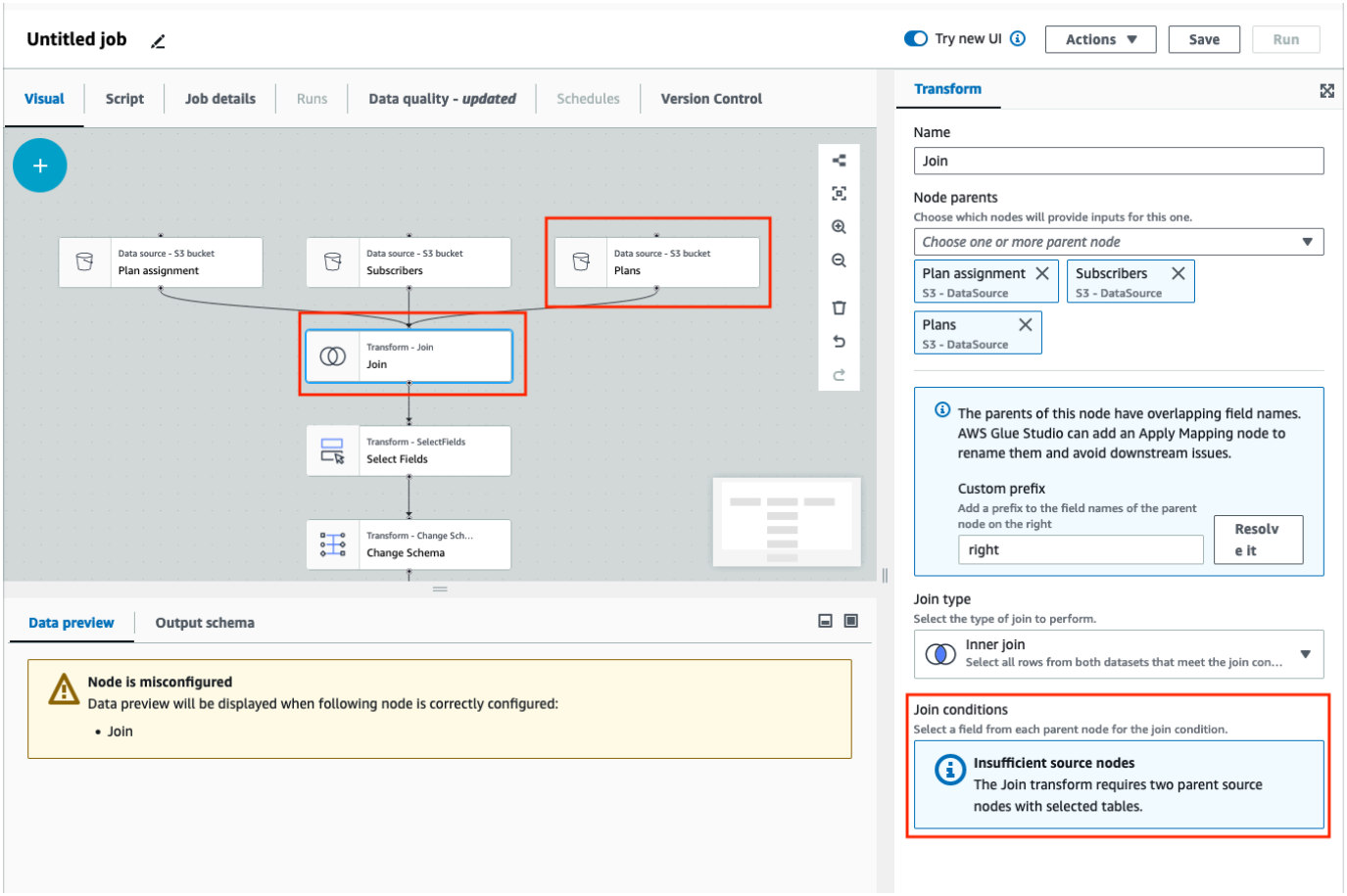


2. Wählen Sie den Knoten aus, den Sie entfernen möchten. Die Arbeitsfläche vergrößert den Knoten. Wählen Sie in der Werkzeugleiste auf der rechten Seite der Leinwand das Papierkorbsymbol. Dadurch wird der Knoten entfernt und jeder Knoten, der mit dem Knoten verbunden ist, wird verschoben und nimmt seinen Platz im Workflow ein. In diesem Beispiel wurde der erste Join-Knoten aus der Arbeitsfläche gelöscht.

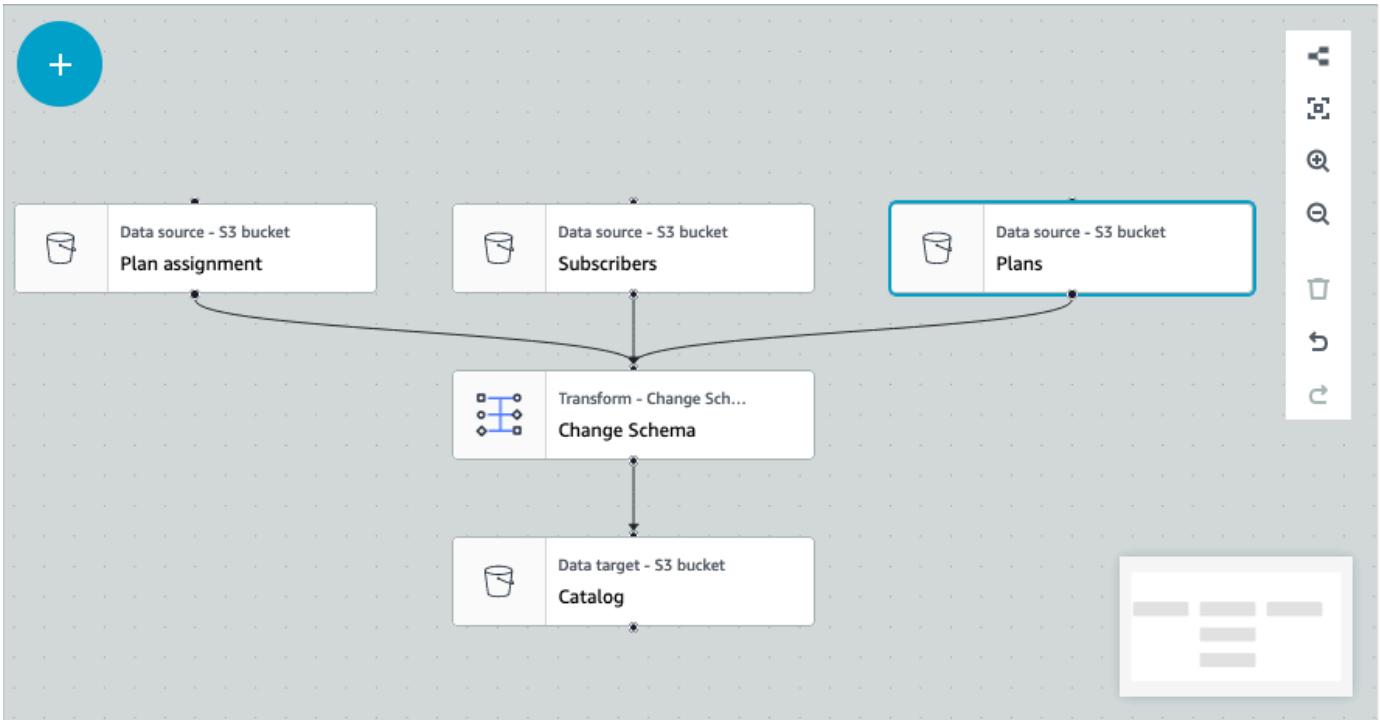
   Wenn Sie einen Knoten im Workflow löschen, AWS Glue werden die Knoten neu angeordnet, sodass sie so organisiert sind, dass kein ungültiger Workflow entsteht. Möglicherweise müssen Sie die Konfiguration eines Knotens dennoch korrigieren.

   In dem Beispiel wurde der Join-Knoten unter dem Abonnenten-Knoten entfernt. Infolgedessen wurde der Quellknoten „Pläne“ auf die oberste Ebene verschoben und ist weiterhin mit dem

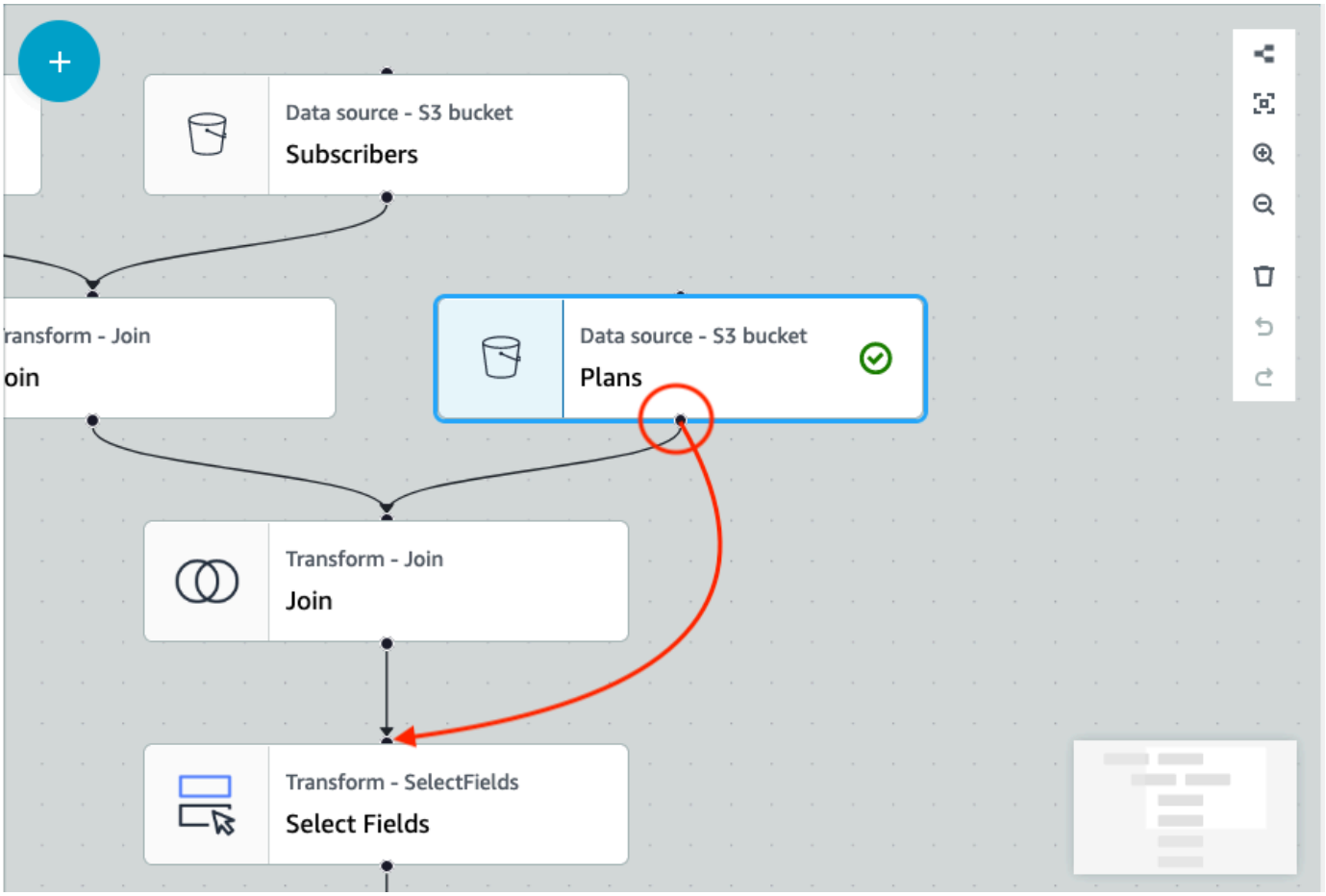
untergeordneten Join-Knoten verbunden. Der Join-Knoten erfordert jetzt eine zusätzliche Konfiguration, da Join zwei übergeordnete Quellknoten mit ausgewählten Tabellen benötigt. Auf der Registerkarte Transformieren rechts neben der Arbeitsfläche wird die fehlende Anforderung unter Join-Bedingungen angezeigt.



3. Löschen Sie den zweiten Join-Knoten und den Knoten Select Fields. Wenn die Knoten gelöscht wurden, sieht der Workflow wie im folgenden Beispiel aus.



4. Um die Knotenverbindungen zu ändern, klicken Sie auf den Griff des Knotens und ziehen Sie die Verbindung auf einen neuen Knoten. Auf diese Weise können Sie Knoten löschen und die Knoten in einem logischen Fluss neu anordnen. In diesem Beispiel wird eine neue Verbindung hergestellt, indem Sie auf den Griff im Knoten Pläne klicken und die Verbindung zum Join-Knoten ziehen, wie durch den roten Pfeil dargestellt.



5. Wenn Sie eine Aktion rückgängig machen müssen, wählen Sie das Rückgängig-Symbol direkt unter dem Papierkorbsymbol in der Werkzeugleiste auf der rechten Seite der Leinwand.

Hinzufügen von Quell- und Zielparametern zum AWS Glue Knoten Datenkatalog

AWS Glue Studio ermöglicht es Ihnen, visuelle Jobs zu parametrisieren. Da die Namen von Katalogtabellen in der Produktions- und in der Entwicklungsumgebung unterschiedlich sein können, können Sie Laufzeitparameter für Datenbanken und Tabellen definieren und auswählen, die ausgeführt werden, wenn Ihr Auftrag ausgeführt wird.

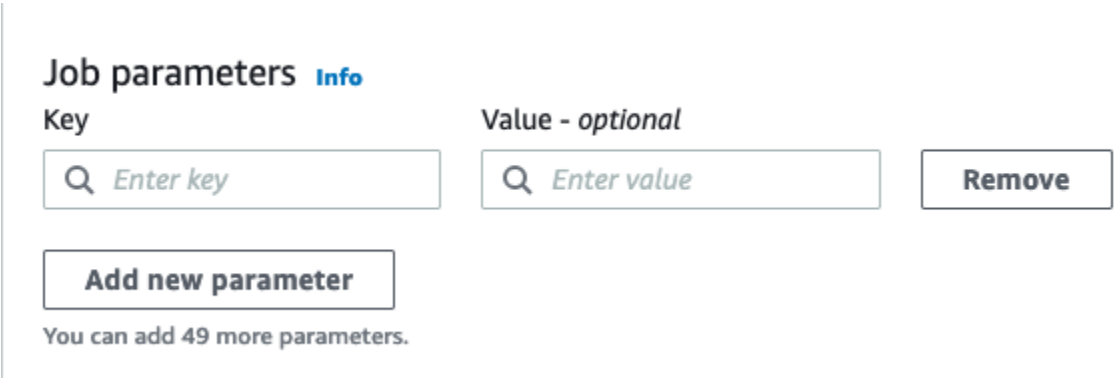
Die Job-Parametrisierung ermöglicht es Ihnen, Quellen und Ziele zu parametrisieren und diese Parameter im Job zu speichern, wenn Sie AWS Glue Knoten „Datenkatalog“. Wenn Sie Quellen und Ziele als Parameter angeben, ermöglichen Sie die Wiederverwendbarkeit von Aufträgen, insbesondere wenn Sie denselben Auftrag in mehreren Umgebungen verwenden. Dies ist nützlich, wenn Sie Code in verschiedenen Bereitstellungsumgebungen bereitstellen. Sie sparen Zeit und

Mühe bei der Verwaltung Ihrer Quellen und Ziele. Darüber hinaus überschreiben die von Ihnen angegebenen benutzerdefinierten Parameter alle Standardargumente für bestimmte Läufe von AWS Glue Jobs.

So fügen Sie Quell- und Zielparameter hinzu

Ob Sie das verwenden AWS Glue Datenkatalogknoten als Quelle oder Ziel. Sie können Laufzeitparameter im Abschnitt Erweiterte Eigenschaften auf der Registerkarte Jobdetails definieren.

1. Wählen Sie das Symbol AWS Glue Datenkatalogknoten entweder als Quell- oder Zielknoten.
2. Wählen Sie die Registerkarte Job details (Auftragsdetails) aus.
3. Wählen Sie Advanced properties (Erweiterte Eigenschaften) aus.
4. Geben Sie im Abschnitt Auftragsparameter einen Schlüsselwert ein. Zum Beispiel wäre `--db.source` der Parameter für eine Datenbankquelle. Sie können einen beliebigen Namen für den Schlüssel eingeben, sofern auf den Schlüsselnamen ein Bindestrich folgt.

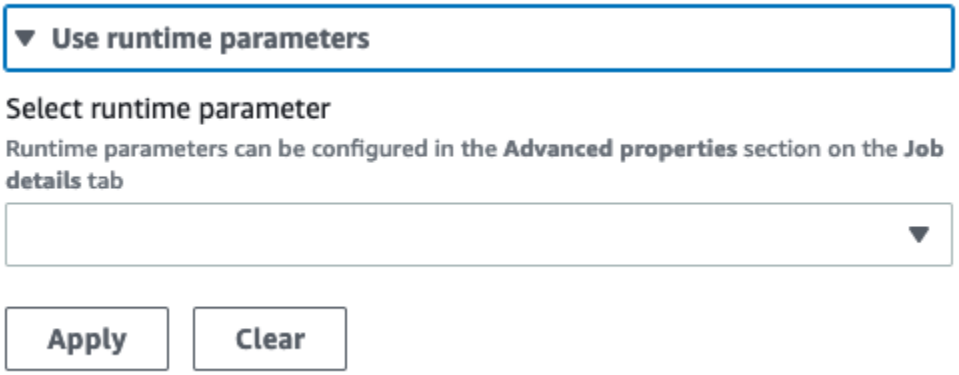


5. Geben Sie den Wert ein. Zum Beispiel wäre `databasename` der Wert für die zu parametrisierende Datenbank.
6. Wählen Sie Add new parameter (Neuen Parameter hinzufügen), wenn Sie weitere Parameter hinzufügen möchten. Max. 50 Parameter sind zulässig. Sobald das Schlüssel-Wert-Paar definiert wurde, können Sie den Parameter in der AWS Glue Knoten Datenkatalog.

So wählen Sie einen Laufzeitparameter aus

> **Note**
>
> Das Verfahren zur Auswahl von Laufzeitparametern für Datenbanken und Tabellen ist derselbe, unabhängig davon, ob AWS Glue Der Datenkatalogknoten ist die Quelle oder das Ziel.

1. Wählen Sie das Symbol AWS Glue Der Datenkatalogknoten ist entweder der Quellknoten oder der Zielknoten.
2. Wählen Sie auf der Registerkarte Data source properties – Data Catalog (Datenquelleneigenschaften – Datenkatalog) unter Database (Datenbank) die Option Use runtime parameters (Laufzeitparameter verwenden) aus.



3. Wählen Sie einen Parameter aus dem Dropdown-Menü. Wenn Sie beispielsweise einen Parameter auswählen, den Sie für eine Quelldatenbank definiert haben, wird die Datenbank automatisch in das Datenbank-Dropdown-Menü eingetragen, wenn Sie Apply (Anwenden) wählen.
4. Wählen Sie im Abschnitt Tabelle einen Parameter aus, den Sie bereits als Quelltabelle definiert haben. Wenn Sie Apply (Anwenden) wählen, wird die Tabelle automatisch als die zu verwendende Tabelle eingetragen.
5. Wenn Sie den Job speichern und ausführen, AWS Glue Studio referenziert während der Auftragsausführung auf die ausgewählten Parameter.

Verwendung von Git-Versionskontrollsystemen in AWS Glue

> **Note**
>
> Notebooks werden derzeit nicht für die Versionskontrolle in unterstützt AWS Glue Studio. Allerdings Versionskontrolle für AWS Glue Jobskripte und visuelle ETL-Jobs werden unterstützt.

Wenn Sie Remote-Repositorys haben und Ihre verwalten möchten AWS Glue Jobs, die Ihre Repositorys verwenden, können Sie verwenden AWS Glue Studio oder um Änderungen AWS CLI an

Ihren Repositorys und Ihren Jobs zu synchronisieren in AWS Glue. Wenn Sie Änderungen auf diese Weise synchronisieren, verschieben Sie den Job von AWS Glue Studio zu Ihrem Repository oder aus dem Repository abrufen zu AWS Glue Studio.

Mit Git-Integration in AWS Glue Studio, du kannst:

- Integrieren Sie in Git-Versionskontrollsysteme wie AWS CodeCommit, GitHub GitLab, und Bitbucket
- Edit (Bearbeiten) AWS Glue Jobs in AWS Glue Studio ob Sie visuelle Jobs oder Skriptjobs verwenden und diese mit einem Repository synchronisieren
- Quellen und Ziele in Aufträgen parametrisieren
- Rufen Sie Jobs aus einem Repository ab und bearbeiten Sie sie in AWS Glue Studio
- Testen Sie Jobs, indem Sie mithilfe branchenübergreifender Workflows Daten aus Branches abrufen und/oder an Branches weiterleiten AWS Glue Studio
- Laden Sie Dateien aus einem Repository herunter und laden Sie Jobs in AWS Glue Studio zur kontenübergreifenden Schaffung von Arbeitsplätzen
- Verwenden Sie das Automatisierungstool Ihrer Wahl (z. B. Jenkins AWS CodeDeploy usw.)

Dieses Video zeigt, wie Sie AWS Glue in Git integrieren und eine kontinuierliche und kollaborative Code-Pipeline aufbauen können.

IAM-Berechtigungen

Stellen Sie sicher, dass der Auftrag über eine der folgenden IAM-Berechtigungen verfügt. Weitere Informationen zum Einrichten von IAM-Berechtigungen finden Sie unter IAM-Berechtigungen einrichten für AWS Glue Studio.

- `AWSGlueServiceRole`
- `AWSGlueConsoleFullAccess`

Für die Git-Integration sind mindestens folgende Aktionen erforderlich:

- `glue:UpdateJobFromSourceControl`— um aktualisieren zu können AWS Glue mit einem Job in einem Versionskontrollsystem
- `glue:UpdateSourceControlFromJob`— um das Versionskontrollsystem mit einem Job aktualisieren zu können, der in gespeichert ist AWS Glue

- `s3:GetObject` – Um das Skript für den Auftrag abrufen zu können, während es an das Versionsverwaltungssystem übergeben wird
- `s3:PutObject` – Um das Skript aktualisieren zu können, wenn ein Auftrag von einem Quellcodeverwaltungssystem abgerufen wird

Voraussetzungen

Um Aufträge in ein Quellcodeverwaltungs-Repository zu übertragen, benötigen Sie:

- ein Repository, das bereits von Ihrem Administrator erstellt wurde
- eine Verzweigung im Repository
- ein persönliches Zugriffstoken (für Bitbucket ist dies das Repository-Zugriffstoken)
- den Benutzernamen des Eigentümers des Repositorys
- setzt die Berechtigungen im Repository auf „Zulassen“ AWS Glue Studio um das Repository zu lesen und in das Repository zu schreiben
  - GitLab— setzt die Token-Gültigkeitsbereiche auf api, read_repository und write_repository
  - Bitbucket – Legen Sie Berechtigungen folgendermaßen fest:
    - Workspace-Mitgliedschaft – Lesen, Schreiben
    - Projekte – Schreiben, Lesen (Administrator)
    - Repositorys – Lesen, Schreiben, Verwalten, Löschen

> **Note**
>
> Bei der Verwendung werden das AWS CodeCommit persönliche Zugriffstoken und der Repository-Besitzer nicht benötigt. Siehe Erste Schritte mit Git und AWS CodeCommit.

Verwenden Sie Jobs aus Ihrem Quellcodeverwaltungs-Repository in AWS Glue Studio

Um einen Job aus Ihrem Quellcodeverwaltungs-Repository abzurufen, der sich nicht in AWS Glue Studio, und um diesen Job zu verwenden in AWS Glue Studio, die Voraussetzungen hängen von der Art des Jobs ab.

Für visuelle Aufträge:

- benötigen Sie einen Ordner und eine JSON-Datei der Auftragsdefinition, die mit dem Auftragsnamen übereinstimmt

  Sehen Sie sich zum Beispiel die folgende Auftragsdefinition an. Die Verzweigung in Ihrem Repository sollte einen Pfad `my-visual-job/my-visual-job.json` enthalten, bei dem sowohl der Ordner als auch die JSON-Datei mit dem Namen des Auftrags übereinstimmen

```
{
  "name" : "my-visual-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-visual-job.py",
    "pythonVersion" : "3"
  },
  "codeGenConfigurationNodes" : "{\"node-nodeID\":{\"S3CsvSource\":
{\"AdditionalOptions\":{\"EnableSamplePath\":false,\"SamplePath\":\"s3://notebook-
test-input/netflix_titles.csv\"},\"Escaper\":\"\",\"Exclusions\":[],\"Name\":\"Amazon
 S3\",\"OptimizePerformance\":false,\"OutputSchemas\":[{\"Columns\":[{\"Name\":
\"show_id\",\"Type\":\"string\"},{\"Name\":\"type\",\"Type\":\"string\"},{\"Name\":
\"title\",\"Type\":\"choice\"},{\"Name\":\"director\",\"Type\":\"string\"},{\"Name\":
\"cast\",\"Type\":\"string\"},{\"Name\":\"country\",\"Type\":\"string\"},{\"Name\":
\"date_added\",\"Type\":\"string\"},{\"Name\":\"release_year\",\"Type\":\"bigint\"},
{\"Name\":\"rating\",\"Type\":\"string\"},{\"Name\":\"duration\",\"Type\":\"string
\"},{\"Name\":\"listed_in\",\"Type\":\"string\"},{\"Name\":\"description\",\"Type
\":\"string\"}]}],\"Paths\":[\"s3://dalamgir-notebook-test-input/netflix_titles.csv
\"],\"QuoteChar\":\"quote\",\"Recurse\":true,\"Separator\":\"comma\",\"WithHeader
\":true}}}"
}
```

Für Skript-Aufträge:

- benötigen Sie einen Ordner, eine JSON-Datei der Auftragsdefinition und das Skript
- der Ordner und die JSON-Datei sollten mit dem Auftragsnamen übereinstimmen. Der Skriptname muss mit dem `scriptLocation` in der Auftragsdefinition übereinstimmen, ebenso die Dateierweiterung

Beispielsweise sollte in der folgenden Auftragsdefinition die Verzweigung in Ihrem Repository einen Pfad `my-script-job/my-script-job.json` und `my-script-job/my-script-job.py` enthalten. Der Skriptname sollte mit dem Namen in der `scriptLocation` einschließlich der Erweiterung des Skripts übereinstimmen

```
{
  "name" : "my-script-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-script-job.py",
    "pythonVersion" : "3"
  }
}
```

Einschränkungen

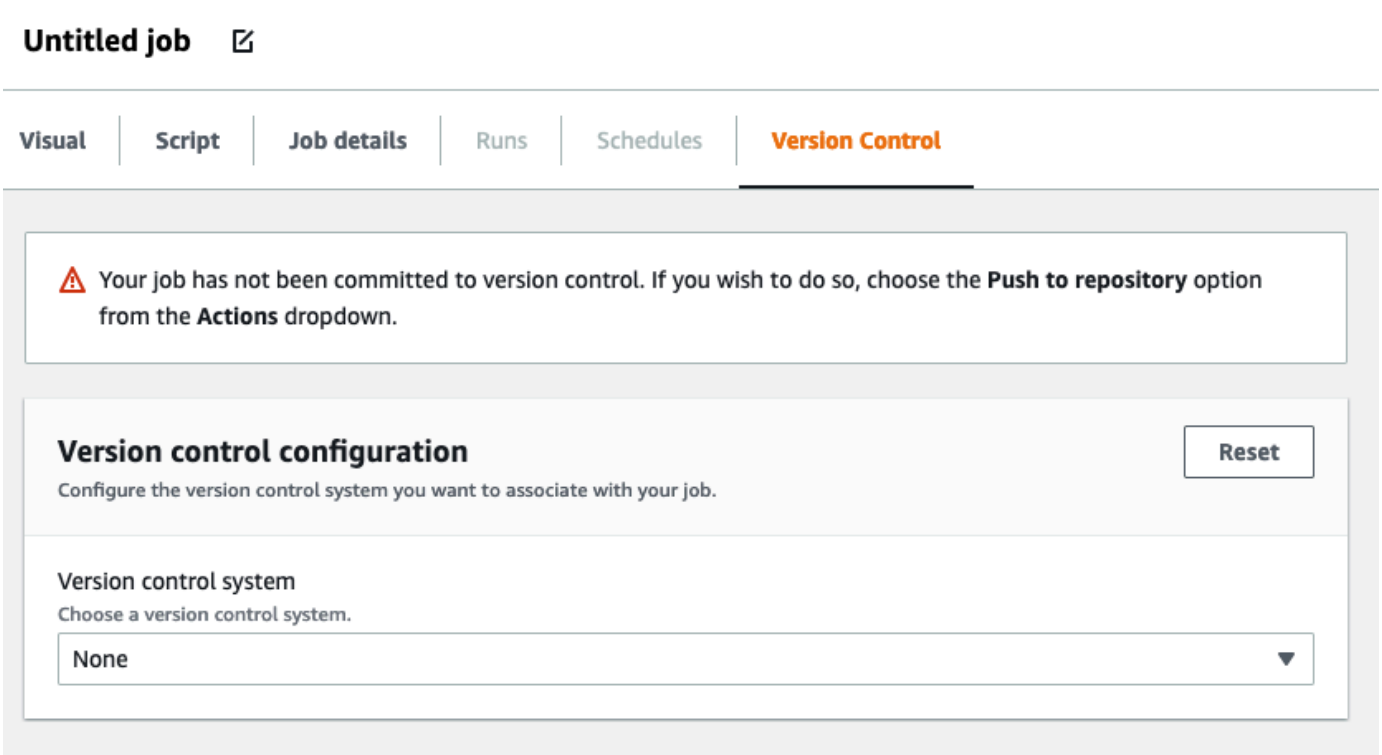
- AWS Glue unterstützt derzeit kein Push/Pull aus GitLab -Groups.

Verbindung von Repositorys zur Versionskontrolle herstellen mit AWS Glue

Sie können die Details Ihres Versionskontroll-Repositorys eingeben und sie auf der Registerkarte Versionskontrolle im AWS Glue Studio Job-Editor. Für die Integration in dein Git-Repository musst du jedes Mal, wenn du dich einloggst, eine Verbindung zu deinem Repository herstellen AWS Glue Studio.

So verbinden Sie ein Git-Versionsverwaltungssystem:

1. In AWS Glue Studio, starten Sie einen neuen Job und wählen Sie den Tab Versionskontrolle.



2. Wählen Sie im Versionsverwaltungssystem den Git-Service aus den verfügbaren Optionen aus, indem Sie auf das Dropdown-Menü klicken.

   - AWS CodeCommit
   - GitHub
   - GitLab
   - Bitbucket

3. Je nachdem, welches Git-Versionsverwaltungssystem Sie auswählen, müssen Sie verschiedene Felder ausfüllen.

   Für AWS CodeCommit:

   Vervollständigen Sie die Repository-Konfiguration, indem Sie das Repository und die Verzweigung für Ihren Auftrag auswählen:

   - Repository — wenn Sie Repositorys in eingerichtet haben AWS CodeCommit, wählen Sie das Repository aus dem Drop-down-Menü aus. Ihre Repositorys werden automatisch in die Liste aufgenommen
   - Branch (Verzweigung) – Wählen Sie die Verzweigung aus dem Dropdown-Menü

- Folder (Ordner) – optional – Geben Sie den Namen des Ordners ein, in dem Ihr Auftrag gespeichert werden soll. Wenn das Feld leer gelassen wird, wird automatisch ein Ordner erstellt. Der Ordnername ist standardmäßig der Auftragsname

Für GitHub:

Vervollständigen Sie die GitHub Konfiguration, indem Sie die Felder ausfüllen:

- Persönliches Zugriffstoken — dies ist das vom GitHub Repository bereitgestellte Token. Weitere Informationen zu persönlichen Zugriffstoken finden Sie in der GitHub Dokumentation
- Repository-Besitzer — das ist der Besitzer des GitHub Repositorys.

Vervollständigen Sie die Repository-Konfiguration, indem Sie das Repository und den Zweig von auswählen GitHub.

- Repository — Wenn Sie Repositorys in eingerichtet haben GitHub, wählen Sie das Repository aus dem Drop-down-Menü aus. Ihre Repositorys werden automatisch in die Liste aufgenommen
- Branch (Verzweigung) – Wählen Sie die Verzweigung aus dem Dropdown-Menü
- Folder (Ordner) – optional – Geben Sie den Namen des Ordners ein, in dem Ihr Auftrag gespeichert werden soll. Wenn das Feld leer gelassen wird, wird automatisch ein Ordner erstellt. Der Ordnername ist standardmäßig der Auftragsname

Für GitLab:

> **Note**
>
> AWS Glue unterstützt derzeit kein Push/Pull aus GitLab -Groups.

- Persönliches Zugriffstoken — das ist das vom Repository bereitgestellte Token. GitLab Weitere Informationen zu persönlichen Zugriffstoken finden Sie unter GitLab Persönliche Zugriffstoken
- Repository-Besitzer — das ist der Besitzer des GitLab Repositorys.

Vervollständigen Sie die Repository-Konfiguration, indem Sie das Repository und den Zweig von auswählen GitLab.

- Repository — Wenn Sie Repositorys in eingerichtet haben GitLab, wählen Sie das Repository aus dem Drop-down-Menü aus. Ihre Repositorys werden automatisch in die Liste aufgenommen
- Branch (Verzweigung) – Wählen Sie die Verzweigung aus dem Dropdown-Menü
- Folder (Ordner) – optional – Geben Sie den Namen des Ordners ein, in dem Ihr Auftrag gespeichert werden soll. Wenn das Feld leer gelassen wird, wird automatisch ein Ordner erstellt. Der Ordnername ist standardmäßig der Auftragsname

Für Bitbucket:

- App-Passwort — Bitbucket verwendet App-Passwörter und keine Repository-Zugriffstoken. Weitere Informationen zu App-Passwörtern findest du unter App-Passwörter.
- Repository-Eigentümer – Dies ist der Eigentümer des Bitbucket-Repositorys. In Bitbucket ist der Eigentümer der Ersteller des Repositorys.

Vervollständigen Sie die Repository-Konfiguration, indem Sie den Arbeitsbereich, das Repository, die Verzweigung und den Ordner von Bitbucket auswählen.

- Arbeitsbereich – Wenn Sie in Bitbucket Arbeitsbereiche eingerichtet haben, wählen Sie den Arbeitsbereich aus dem Dropdown-Menü aus. Ihre Arbeitsbereiche werden automatisch mit Daten gefüllt
- Repository – Wenn Sie in Bitbucket Repositorys eingerichtet haben, wählen Sie das Repository aus dem Dropdown-Menü aus. Ihre Repositorys werden automatisch mit Daten gefüllt
- Verzweigung – Wählen Sie die Verzweigung aus dem Dropdown-Menü aus. Ihre Verzweigungen werden automatisch mit Daten gefüllt
- Folder (Ordner) – optional – Geben Sie den Namen des Ordners ein, in dem Ihr Auftrag gespeichert werden soll. Wenn das Feld leer gelassen wird, wird automatisch ein Ordner mit dem Auftragsnamen erstellt.

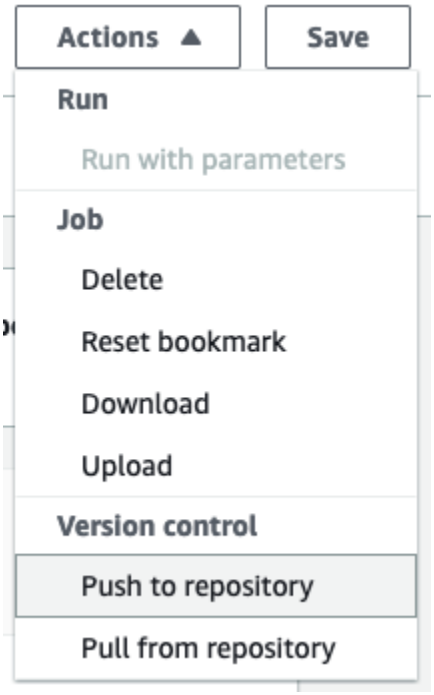
4. Wählen Sie oben auf der Seite Speichern aus AWS Glue Studio Auftrag

Pushing AWS Glue Jobs in das Quell-Repository

Sobald Sie die Details Ihres Versionskontrollsystems eingegeben haben, können Sie Jobs in bearbeiten AWS Glue Studio und übertragen Sie die Jobs in Ihr Quell-Repository. Wenn Sie mit Git-Konzepten wie Übertragen und Ziehen nicht vertraut sind, lesen Sie dieses Tutorial unter Erste Schritte mit Git und AWS CodeCommit.

Um Ihren Auftrag an ein Repository zu übergeben, müssen Sie die Details Ihres Versionsverwaltungssystems eingeben und Ihren Auftrag speichern.

1. Im AWS Glue StudioJob, wählen Sie Aktionen. Dadurch werden zusätzliche Menüoptionen geöffnet.



2. Wählen Sie Push to repository (An das Repository übergeben) aus.

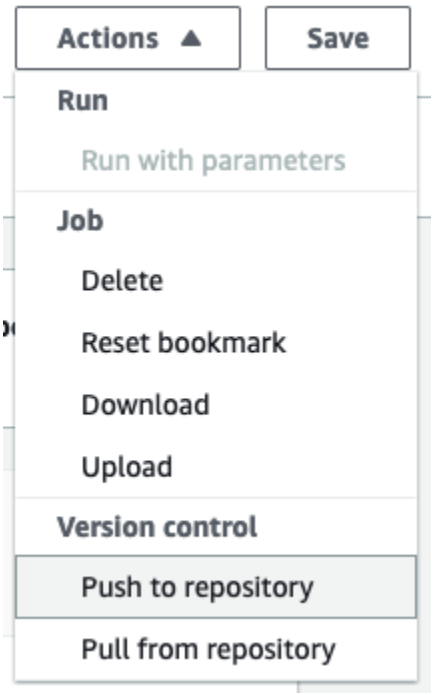
   Durch diese Aktion wird der Auftrag gespeichert. Wenn Sie zum Repository pushen, AWS Glue Studio überträgt die zuletzt gespeicherte Änderung. Wenn der Job im Repository von Ihnen oder einem anderen Benutzer geändert wurde und nicht mehr mit dem Job in synchronisiert ist AWS Glue Studio, wird der Job im Repository mit dem Job überschrieben, der in gespeichert ist AWS Glue Studio wenn Sie den Job von pushen AWS Glue Studio.

3. Wählen Sie Confirm (Bestätigen) um die Aktion abzuschließen. Dadurch wird ein neues Commit im Repository erstellt. Wenn Sie verwenden AWS CodeCommit, wird in einer Bestätigungsnachricht ein Link zum letzten Commit angezeigt AWS CodeCommit.

Ziehend AWS Glue Jobs aus dem Quell-Repository

Sobald Sie die Details Ihres Git-Repositorys in den Tab Versionskontrolle eingegeben haben, können Sie auch Jobs aus Ihrem Repository abrufen und bearbeiten in AWS Glue Studio.

1. Im AWS Glue Studio Job, wähle Aktionen. Dadurch werden zusätzliche Menüoptionen geöffnet.



2. Wählen Sie Pull from repository (Aus dem Repository ziehen) aus.
3. Wählen Sie Bestätigen aus. Dies nimmt den neuesten Commit aus dem Repository und aktualisiert Ihren Job in AWS Glue Studio.
4. Bearbeiten Sie Ihren Job in AWS Glue Studio. Wenn Sie Änderungen vornehmen, können Sie Ihren Job mit Ihrem Repository synchronisieren, indem Sie im Drop-down-Menü Aktionen die Option An Repository übertragen wählen.

Code erstellen mit AWS Glue Studio notebooks

Dateningenieure können Inhalte verfassen AWS Glue Jobs schneller und einfacher als zuvor mithilfe der interaktiven Notebook-Oberfläche in AWS Glue Studio oder interaktive Sitzungen in AWS Glue.

Einschränkungen

- AWS Glue Studio Notebooks unterstützen Scala nicht.

Themen

- Übersicht über die Verwendung von Notebooks
- Erstellen eines ETL-Jobs mithilfe von Notebooks in AWS Glue Studio
- Notebook-Editor-Komponenten
- Speichern von Notebook und Auftragsskript
- Verwalten von Notebooksitzungen
- Amazon Q Developer mit AWS Glue Studio Notebooks verwenden

Übersicht über die Verwendung von Notebooks

AWS Glue Studio ermöglicht das interaktive Verfassen von Jobs in einer Notebook-Oberfläche, die auf Jupyter Notebooks basiert. Über Notizbücher in AWS Glue Studio, Sie können Jobskripte bearbeiten und die Ausgabe anzeigen, ohne einen vollständigen Job ausführen zu müssen, und Sie können den Datenintegrationscode bearbeiten und die Ausgabe anzeigen, ohne einen vollständigen Job ausführen zu müssen, und Sie können Markdown hinzufügen und Notizbücher als.ipynb-Dateien und Jobskripts speichern. Sie können ein Notebook starten, ohne Software lokal zu installieren oder Server zu verwalten. Wenn Sie mit Ihrem Code zufrieden sind, AWS Glue Studio kann Ihr Notizbuch mit einem Klick in einen Glue-Job umwandeln.

Die Nutzung von Notebooks bietet unter anderem folgende Vorteile:

- Kein Cluster zum Bereitstellen oder Verwalten
- Keine gebührenpflichtiger Leerlauf-Cluster
- Keine Vorabkonfiguration erforderlich
- Keine Installation von Jupyter Notebooks erforderlich
- Dieselbe Laufzeit/Plattform wie AWS Glue ETL

Wenn Sie ein Notizbuch starten durch AWS Glue Studio, alle Konfigurationsschritte werden für Sie erledigt, sodass Sie Ihre Daten untersuchen und nach nur wenigen Sekunden mit der Entwicklung Ihres Job-Skripts beginnen können. AWS Glue Studio konfiguriert ein Jupyter-Notebook mit dem AWS Glue Jupyter-Kernel. Sie müssen keine Netzwerkverbindungen oder Entwicklungsendpunkte konfigurieren VPCs, um dieses Notebook verwenden zu können.

So erstellen Sie Aufträge mit der Notebook-Schnittstelle:

- konfigurieren Sie die erforderlichen IAM-Berechtigungen.

- eine Notebook-Sitzung starten, um einen Auftrag zu erstellen
- Code in die Zellen im Notebook schreiben
- Code ausführen und testen, um die Ausgabe anzuzeigen
- Speichern des Auftrags

Nachdem Ihr Notizbuch gespeichert wurde, ist Ihr Notizbuch voll AWS Glue Arbeit. Sie können alle Aspekte des Auftrags verwalten, z. B. das Planen von Auftragsläufen, das Festlegen von Auftragsparametern und das Anzeigen des Auftragslaufverlaufs direkt neben Ihrem Notebook.

Erstellen eines ETL-Jobs mithilfe von Notebooks in AWS Glue Studio

Um mit der Verwendung von Notebooks in der AWS Glue Studio Konsole

1. Hängen Sie AWS Identity and Access Management Richtlinien an das an AWS Glue Studio Benutzer und erstellen Sie eine IAM-Rolle für Ihren ETL-Job und Ihr ETL-Notebook.
2. Konfigurieren Sie zusätzliche IAM-Sicherheit für Notebooks, wie unter Erteilen von Berechtigungen für die IAM-Rolle beschrieben.
3. Öffnen Sie AWS Glue Studio Konsole unter. https://console.aws.amazon.com/gluestudio/

   > **Note**
   >
   > Vergewissern Sie sich, dass Ihr Browser Cookies von Drittanbietern nicht blockiert. Jeder Browser, der Cookies von Drittanbietern standardmäßig oder per Benutzereinstellung blockiert, verhindert das Starten von Notebooks. Weitere Informationen zum Verwalten von Cookies finden Sie unter:

   - Chrome
   - Firefox
   - Safari
4. Wählen Sie das Symbol Jobs (Aufträge) im Navigationsmenü der linken Seite.
5. Klicken Sie auf Jupyter Notebook und dann auch Create (Erstellen), um eine neue Notebook-Sitzung zu starten.
6. Auf der Seite Create job in Jupyter notebook (Auftrag im Jupyter-Notebook erstellen) geben Sie den Auftragsnamen und die zu verwendende IAM-Rolle an. Wählen Sie Job erstellen aus.

Nach kurzer Zeit erscheint der Notebook-Editor.

7. Nachdem Sie den Code hinzugefügt haben, müssen Sie die Zelle ausführen, um eine Sitzung zu initiieren. Es gibt mehrere Möglichkeiten, die Zelle auszuführen:

   - Drücken Sie die Wiedergabetaste.
   - Verwenden Sie die Tastenkombination:
     - Auf MacOS, Command (Befehl) + Enter (Eingabe) um die Zelle zu auszuführen.
     - Auf Windows Shift (Umschalttaste) + Enter (Eingabe) um die Zelle auszuführen.

   Informationen zum Schreiben von Code über eine Jupyter-Notebook-Schnittstelle finden Sie unter The Jupyter Notebook User Documentation (Die Jupyter Notebook-Benutzerdokumentation).
8. Um Ihr Skript zu testen, führen Sie das gesamte Skript oder einzelne Zellen aus. Jede Befehlsausgabe wird im Bereich unter der Zelle angezeigt.
9. Nachdem Sie mit der Entwicklung Ihres Notebooks fertig sind, können Sie den Auftrag speichern und dann ausführen. Sie finden das Skript auf der Registerkarte Script (Skript). Jegliche Magie, die Sie dem Notizbuch hinzugefügt haben, wird entfernt und nicht als Teil des generierten Skripts gespeichert AWS Glue Arbeit. AWS Glue Studio fügt am Ende Ihres generierten Skripts automatisch ein `job.commit()` aus dem Notizbuchinhalt hinzu.

   Weitere Informationen zum Ausführen eines Auftrags finden Sie unter Starten einer Auftragsausführung.

Notebook-Editor-Komponenten

Die Oberfläche des Notebook-Editors enthält die folgenden Hauptabschnitte.

- Notebook-Schnittstelle (Hauptbereich) und Symbolleiste
- Registerkarten zur Auftragsbearbeitung

Der Notebook-Editor

Das Tool AWS Glue Studio Der Notebook-Editor basiert auf der Jupyter Notebook Application. Das Tool AWS Glue Studio Die Notebook-Oberfläche ähnelt der von Juypter Notebooks bereitgestellten,

die im Abschnitt Notebook-Benutzeroberfläche beschrieben wird. Das von interaktiven Sitzungen verwendete Notebook ist ein Jupyter Notebook.

Obwohl die AWS Glue Studio Notebook ähnelt Juptyer Notebooks, unterscheidet sich jedoch in einigen wesentlichen Punkten:

- derzeit ist das AWS Glue Studio Notebook kann keine Erweiterungen installieren
- Sie können nicht mehrere Registerkarten verwenden; es besteht eine 1:1 -Beziehung zwischen einem Job und einem Notebook
- die AWS Glue Studio Notebook hat nicht dasselbe oberste Dateimenü wie in Jupyter Notebooks
- derzeit ist das AWS Glue Studio Notebook läuft nur mit dem AWS Glue Kernel. Beachten Sie, dass Sie den Kernel nicht selbst aktualisieren können.

AWS Glue Studio Registerkarten zur Auftragsbearbeitung

Die Registerkarten, mit denen Sie mit dem ETL-Auftrag interagieren, befinden sich oben auf der Notebook-Seite. Sie ähneln den Tabs, die im visuellen Job-Editor von angezeigt werden AWS Glue Studio, und sie führen dieselben Aktionen aus.

- Notebook – Über diese Registerkarte können Sie das Job-Skript über die Notebook-Schnittstelle anzeigen.
- Job details (Auftragsdetails) – Konfigurieren Sie die Umgebung und die Eigenschaften für die Auftragsausführungen.
- Runs (Ausführungen) – Zeigen Sie Informationen über frühere Läufe dieses Auftrags an.
- Schedules (Zeitpläne) – Konfigurieren Sie einen Zeitplan für die Ausführung Ihres Auftrags zu bestimmten Zeiten.

Speichern von Notebook und Auftragsskript

Sie können Ihr Notebook und das Auftragsskript, das Sie erstellen, jederzeit speichern. Wählen Sie einfach die Schaltfläche Save (Speichern) in der oberen rechten Ecke, genauso wie wenn Sie den visuellen oder den Skript-Editor verwenden würden.

Wenn Sie Save (Speichern) auswählen wird die Notebook-Datei auf den Standardspeicherorten gespeichert:

- Standardmäßig wird das Auftragsskript am Amazon-S3-Speicherort gespeichert, der im Job Details-Tab (Auftragsdetails-Tab) unter Advanced properties (Erweiterte Eigenschaften) in der Auftragsdetails-Eigenschaft Script path (Skript-Pfad) angegeben ist. Auftragsskripts werden in einem Unterordner mit dem Namen `Scripts` gespeichert.
- Standardmäßig wird die Notebook-Datei (`.ipynb`) an dem Amazon S3-Standort gespeichert, der im Job Details-Tab (Auftragsdetails-Tab) unter Advanced properties (Erweiterte Eigenschaften), in den Jobdetails Script-Path (Skript-Pfad) angezeigt wird. Notebook-Dateien werden in einem Unterordner mit dem Namen `Notebooks` gespeichert.

> **Note**
>
> Wenn Sie den Auftrag speichern, enthält das Auftragsskript nur die Codezellen aus dem Notebook. Die Markdown-Zellen und Magics sind nicht im Auftrags-Skript enthalten. Allerdings enthält die `.ipynb`-Datei alle Markdown und Magics.

Nachdem Sie den Auftrag gespeichert haben, können Sie den Auftrag dann mit dem Skript ausführen, das Sie im Notebook erstellt haben.

Verwalten von Notebooksitzungen

Notebooks in AWS Glue Studio basieren auf dem interaktiven Sitzungs-Feature von AWS Glue. Es fallen Kosten für die Verwendung interaktiver Sitzungen an. Um Ihre Kosten zu verwalten, können Sie die für Ihr Konto erstellten Sitzungen überwachen und die Standardeinstellungen für alle Sitzungen konfigurieren.

Ändern Sie das Standard-Timeout für alle Notebook-Sitzungen

Standardmäßig wird das bereitgestellte AWS Glue Studio-Notebook nach 12 Stunden deaktiviert, wenn das Notebook gestartet und keine Zellen ausgeführt wurden. Es sind keine Kosten damit verbunden und das Timeout ist nicht konfigurierbar.

Sobald Sie eine Zelle ausgeführt haben, wird eine interaktive Sitzung gestartet. Diese Sitzung hat standardmäßig ein Timeout von 48 Stunden. Dieses Timeout kann durch die Übergabe eines `%idle_timeout`-Magic-Befehls konfiguriert werden, bevor eine Zelle ausgeführt wird.

Ändern des Standardsitzungstimeouts für Notebooks in AWS Glue Studio

1. Geben Sie im Notebook `%idle_timeout`-Magic in einer Zelle ein und geben Sie den Timeout-Wert in Minuten an.
2. Beispiel: `%idle_timeout 15` ändert das Standard-Timeout zu 15 Minuten. Wenn die Sitzung 15 Minuten nicht verwendet wird, wird die Sitzung automatisch gestoppt.

Installieren zusätzlicher Python-Module

Wenn Sie mit pip zusätzliche Module zu Ihrer Sitzung installieren möchten, können Sie dies mit `%additional_python_modules` machen, um diese Ihrer Sitzung hinzuzufügen:

```
%additional_python_modules awswrangler, s3://amzn-s3-demo-bucket/mymodule.whl
```

Alle Argumente für additional_python_modules werden an übergeben `pip3 install -m <>`

Eine Liste der verfügbaren Python-Module finden Sie unter Verwenden von Python-Bibliotheken mit AWS Glue.

Ändern der AWS Glue-Konfiguration

Sie können Magics verwenden, um die Werte der Auftragskonfiguration von AWS Glue zu kontrollieren. Wenn Sie einen Auftragskonfigurationswert ändern möchten, müssen Sie die richtigen Magics im Notebook verwenden. Lesen Sie Unterstützte Magics in AWS Glue-interaktiven Sitzungen für Jupyter.

> **Note**
>
> Das Überschreiben von Eigenschaften für eine ausgeführte Sitzung ist nicht mehr möglich. Um die Konfigurationen der Sitzung zu ändern, können Sie die Sitzung beenden, die neuen Konfigurationen festlegen und dann eine neue Sitzung starten.

AWS Glue unterstützt verschiedene Worker-Typen. Sie können den Worker-Typ mit `%worker_type` einstellen. Beispiel: `%worker_type G.2X` . Zu den verfügbaren Arbeitertypen gehören G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X und R.8X. Der Standardwert ist G.1X.

Sie können auch die Anzahl der Mitarbeiter mit `%number_of_workers` angeben. Um beispielsweise 40 Mitarbeiter anzugeben: `%number_of_workers 40`.

Weitere Informationen finden Sie unter Defining Job Properties (Definieren der Auftragseigenschaften)

Anhalten einer Notebooksitzung

Verwenden Sie die -Magic, um eine Notebook-Sitzung zu beenden `%stop_session`.

Wenn Sie das Notizbuch in der AWS Konsole verlassen, erhalten Sie eine Warnmeldung, in der Sie wählen können, ob Sie die Sitzung beenden möchten.

Amazon Q Developer mit AWS Glue Studio Notebooks verwenden

AWS Glue Studio ermöglicht es Ihnen, Aufträge interaktiv in einer Notebook-Schnittstelle basierend auf Jupyter Notebooks zu erstellen. Die Verwendung von Amazon Q Developer verbessert das Authoring-Erlebnis in AWS Glue Studio Notizbüchern.

Die Amazon Q Developer-Erweiterung unterstützt das Schreiben von Code, indem sie Codeempfehlungen generiert und Verbesserungen im Zusammenhang mit Codeproblemen vorschlägt. Amazon Q Developer unterstützt sowohl Python als auch Scala, die beiden Sprachen, die für die Codierung von ETL-Skripts für Spark-Jobs in AWS Glue Studio Notebooks verwendet werden.

Was ist Amazon Q Developer?

Amazon Q Developer ist ein Service, der auf maschinellem Lernen basiert und die Produktivität von Entwicklern verbessert. Amazon Q Developer erreicht dies, indem es Codeempfehlungen generiert, die auf den Kommentaren der Entwickler in natürlicher Sprache und ihrem Code in der IDE basieren. Der Service lässt sich in Amazon SageMaker AI Studio JupyterLab, Amazon SageMaker AI Notebook-Instances und andere integrierte Entwicklungsumgebungen integrieren ()IDEs.

Weitere Informationen finden Sie unter Amazon Q Developer verwenden mit AWS Glue Studio.

Status der AWS Glue-Auftragsausführung in der Konsole

Sie können den Status eines AWS Glue-ETL-Auftrags (Extract, Transform, Load (ETL) anzeigen, während er ausgeführt wird oder nachdem er angehalten wurde. Sie können den Status mit der AWS Glue-Konsole anzeigen.

Zugriff auf das Dashboard für die Auftragsüberwachung

Sie greifen auf das Job-Monitoring-Dashboard zu, indem Sie im AWS Glue Navigationsbereich unter ETL-Jobs auf den Link Job Run Monitoring klicken.

Übersicht über das Dashboard zur Auftragsüberwachung

Das Dashboard zur Auftragsüberwachung bietet eine Gesamtübersicht der Auftragsausführungen und gibt an, wie viele Aufträge sich jeweils in den Status Running (Wird ausgeführt), Canceled (Abgebrochen), Success (Erfolgreich) und Failed (Fehlgeschlagen) befinden. Zusätzliche Kacheln zeigen die Gesamt-Erfolgsquote der Auftragsausführung, die geschätzte DPU-Auslastung für Aufträge, eine Aufschlüsselung der Auftragsstatus nach Auftragstyp, Worker-Typ und Datum.

Die Diagramme in den Kacheln sind interaktiv. Sie können einen beliebigen Block in einem Diagramm auswählen und einen Filter anwenden, sodass nur diese Aufträge in der Tabelle Job runs (Auftragsausführungen) unten auf der Seite angezeigt werden.

Sie können den Datumsbereich der auf dieser Seite angezeigten Informationen mithilfe der Auswahl Date range (Datumsbereich) ändern. Wenn Sie den Datumsbereich ändern, werden auf den Informationskacheln nur die Werte aus den vergangenen x Tagen angezeigt. Sie können auch einen bestimmten Datumsbereich anwenden, wenn Sie Custom (Benutzerdefiniert) bei der Datumsbereichauswahl auswählen.

Anzeigen von Auftragsausführungen

> **Note**
>
> Der Verlauf der Auftragsausführung ist 90 Tage lang für Ihren Workflow und Ihre Auftragsausführung zugänglich.

Die Ressource Job runs (Auftragsausführungen) zeigt die Aufträge für den angegebenen Datumsbereich und die Filter an.

Sie können die Aufträge nach zusätzlichen Kriterien filtern, z. B. Status, Worker-Typ, Auftragstyp und Auftragsname. Im Filterfeld oberhalb der Tabelle können Sie den Text eingeben, der als Filter verwendet werden soll. Während Sie den Text eingeben, werden die Tabellenergebnisse mit Zeilen ergänzt, die übereinstimmenden Text enthalten.

Sie können eine Teilmenge der Aufträge anzeigen, indem Sie Elemente aus den Diagrammen im Dashboard zur Auftragsüberwachung auswählen. Wenn Sie beispielsweise die Anzahl der laufenden Aufträge in der Kachel Job runs summary (Zusammenfassung der Auftragsausführungen) auswählen, dann zeigt die Liste Auftragsausführungen nur die Aufträge an, die derzeit den Status `Running` (Wird ausgeführt) haben. Wenn Sie einen der Balken im Balkendiagramm Worker type breakdown (Gliederung des Worker-Typs) auswählen, werden nur Auftragsausführungen mit dem übereinstimmenden Worker-Typ und Status in der Liste Job runs (Auftragsausführungen) angezeigt.

In der Ressourcenliste Job runs (Auftragsausführungen) sind die Details für die Auftragsausführungen zu sehen. Sie können die Zeilen in der Tabelle sortieren, indem Sie eine Spaltenüberschrift auswählen. In der Tabelle finden Sie die folgenden Informationen:

Property (Eigenschaft)	Description (Beschreibung)
Job name (Auftragsname)	Der Name des -Auftrags.
Type (Typ)	Den Typen der Auftragsumgebung. • Glue ETL: Wird in einer von AWS Glue verwalteten Apache-Spark-Umgebung ausgeführt. • Glue Streaming: Läuft in einer Apache-Spark-Umgebung und führt ETL für Datenströme aus. • Python-Shell: Führt Python-Skripte als Shell aus.
Start time (Startzeit)	Das Datum und die Uhrzeit, an denen diese Auftragsausführung gestartet wurde
End time (Endzeit)	Das Datum und die Uhrzeit, an denen diese Auftragsausführung abgeschlossen wurde
Run status (Ausführungsstatus)	Den aktuellen Status der Auftragsausführung. Der Status kann die folgenden Werte haben: • `STARTING` • `RUNNING`

Property (Eigenschaft)	Description (Beschreibung)
	• `STOPPING` • `STOPPED` • `SUCCEEDED` • `FAILED` • `TIMEOUT`
Run time (Laufzeit)	Die Zeit, in der durch die Auftragsausführung Ressourcen verbraucht wurden
Capacity (Kapazität)	Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die für diesen Auftragslauf zugewiesen wurden. Weitere Hinweise zur Kapazitätsplanung finden Sie unter Überwachung für die DPU-Kapazitätsplanung im AWS Glue-Entwicklerhandbuch.

Property (Eigenschaft)	Description (Beschreibung)
Worker type (Worker-Typ)	Der Typ des vordefinierten Workers, der zugeordnet wurde, als ein Auftrag in der Ausführung war. Die Werte können `G.1X`, `G.2X`, `G.4X` oder `G.8X` sein. • **`G.1X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 84 GB Festplatte (ca. 34 GB freier Speicher) zugeordnet. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge. Dies ist die Standardeinstellung Worker-Typ für Aufträge mit AWS Glue-Version 2.0 oder höher • **`G.2X`** – Wenn Sie diese Art auswählen , müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 128 GB Festplatte (ca. 77 GB freier Speicherplatz) zu. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge und solche, die Machine-Learning-Transformationen ausführen. • **`G.4X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte (ca. 235 GB freier Speicherplatz) zu. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Aufträge der AWS Glue-Version

Property (Eigenschaft)	Description (Beschreibung)
	3.0 oder höher in den folgenden AWS -Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland) und Europa (Stockholm). • **`G.8X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte (ca. 487 GB freier Speicher) zu. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Jobs der AWS Glue Version 3.0 oder höher in denselben AWS Regionen verfügbar, die für den Worker-Typ unterstützt werden. `G.4X`
DPU hours (DPU-Stunden)	Die geschätzte Anzahl der für die Ausführung des Jobs DPUs verwendeten. Eine DPU ist ein relatives Maß für die Rechenleistung. DPUs werden verwendet, um die Kosten für die Ausführung Ihres Jobs zu ermitteln. Weitere Informationen finden Sie in der AWS Glue-Preisliste.

Sie können eine beliebige Auftragsausführung in der Liste auswählen und sich weitere Informationen anzeigen lassen. Wählen Sie eine Auftragsausführung aus und führen Sie dann einen der folgenden Schritte aus:

- Wählen Sie im Menü Actions (Aktionen) die Option View job (Auftrag anzeigen), um den Auftrag im visuellen Editor zu sehen.
- Wählen Sie im Menü Actions (Aktionen) die Option Stop run (Ausführung stoppen), um die aktuelle Ausführung des Auftrags zu stoppen.
- Wählen Sie die Schaltfläche „ CloudWatch Protokolle anzeigen“, um die Job-Ausführungsprotokolle für diesen Job anzuzeigen.
- Wählen Sie Details anzeigen aus, um die Seite mit den Details zur Auftragsausführung zu öffnen.

Anzeigen der Auftragsausführungsprotokolle

Sie können die Auftragsprotokolle auf verschiedene Weise anzeigen:

- Wählen Sie auf der Seite Überwachung in der Tabelle Auftragsausführungen eine Auftragsausführung aus, und klicken Sie dann auf CloudWatch Protokolle anzeigen.
- Wählen Sie im visuellen Auftragseditor auf der Registerkarte Runs (Ausführungen) eines Auftrags die Hyperlinks aus, um die Protokolle anzuzeigen:
  - Logs (Protokolle) – Link zu den Apache-Spark-Auftragsprotokollen, die geschrieben werden, wenn die kontinuierliche Protokollierung für eine Auftragsausführung aktiviert ist. Wenn Sie diesen Link wählen, gelangen Sie zu den Amazon CloudWatch Protokollen in der `/aws-glue/jobs/logs-v2` Protokollgruppe. Standardmäßig enthalten die Protokolle keine unwichtigen Meldungen von Apache-Hadoop-YARN-Heartbeats und Apache-Spark-Treiber- oder Executor-Protokollmeldungen. Weitere Informationen zur fortlaufenden Protokollierung finden Sie unter Kontinuierliche Protokollierung für AWS Glue-Aufträge im AWS Glue-Entwicklerhandbuch.
  - Error logs (Fehlerprotolle) – Link zu den Protokollen, die für diese Auftragsausführung in `stderr` geschrieben werden. Wenn Sie diesen Link auswählen, gelangen Sie zu den Amazon CloudWatch -Protokollen in der Protokollgruppe `/aws-glue/jobs/error`. Die Protokolle zeigen Details zu den einzelnen aufgetretenen Fehlern.
  - Logs (Protokolle) – Link zu den Protokollen, die für diese Auftragsausführung in `stdout` geschrieben wurden. Wenn Sie diesen Link auswählen, gelangen Sie zu den Amazon CloudWatch -Protokollen in der Protokollgruppe `/aws-glue/jobs/output`. In diesen Protokolle sehen Sie alle Details über die Tabellen, die im AWS Glue Data Catalog erstellt wurden und alle aufgetretenen Fehler.

Anzeigen der Details einer Auftragsausführung

Sie können einen Auftrag in der Liste Job runs (Auftragsausführungen) auf der Seite Monitoring (Überwachung) auswählen und sich dann mit View run details (Details zu Ausführung anzeigen) detaillierte Informationen für diese Ausführung des Auftrags anzeigen lassen.

Die Informationen auf der Seite mit den Ausführungsdetails umfassen Folgendes:

Property (Eigenschaft)	Description (Beschreibung)
Job name (Auftragsname)	Der Name des -Auftrags.
Run status (Ausführungsstatus)	Den aktuellen Status der Auftragsausführung. Der Status kann die folgenden Werte haben: • `STARTING` • `RUNNING` • `STOPPING` • `STOPPED` • `SUCCEEDED` • `FAILED` • `TIMEOUT`
Glue version (Glue-Version)	Die AWS Glue-Version, die von der Auftragsausführung verwendet wird.
Recent attempt (Letzte Versuche)	Die Anzahl der automatischen Wiederholungsversuche für diese Auftragsausführung.
Start time (Startzeit)	Das Datum und die Uhrzeit, an denen diese Auftragsausführung gestartet wurde
End time (Endzeit)	Das Datum und die Uhrzeit, an denen diese Auftragsausführung abgeschlossen wurde
Startupzeit	Dauer der Vorbereitung für die Auftragsausführung.

Property (Eigenschaft)	Description (Beschreibung)
Execution time (Ausführungszeit)	Dauer der Ausführung des Auftragsskripts.
Trigger name (Auslösername)	Der Name des Triggers, der dem Auftrag zugeordnet ist.
Last modified on (Letzte Änderung)	Das Datum, an dem der Auftrag zuletzt geändert wurde.
Security configuration (Sicherheitskonfiguration)	Die Sicherheitskonfiguration für den Job, die Amazon S3 S3-Verschlüsselungs- und CloudWatch Verschlüsselungseinstellungen für Job-Lesezeichen umfasst.
Zeitüberschreitung	Der Schwellenwert für eine Zeitüberschreitung bei der Auftragsausführung.
Allocated capacity (Zugewiesene Kapazität)	Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die für diese Auftragsausführung zugewiesen wurden. Weitere Hinweise zur Kapazitätsplanung finden Sie unter Überwachung für die DPU-Kapazitätsplanung im AWS Glue-Entwicklerhandbuch.
Max capacity (Maximale Kapazität)	Die maximale Kapazität, die für die Auftragsausführung verfügbar ist.
Number of workers (Anzahl der Worker)	Die Anzahl der Worker, die für die Auftragsausführung verwendet werden.

Property (Eigenschaft)	Description (Beschreibung)
Worker type (Worker-Typ)	Der Typ der vordefinierten Worker, die der Auftragsausführung zugewiesen sind Werte können `G.1X` oder `G.2X` sein. • **`G.1X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist einer DPU (4 VCPUs, 16 GB Arbeitsspeicher, 64 GB Festplatte) zugeordnet und stellt einen Executor pro Worker bereit. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge. Dies ist die Standardeinstellung Worker-Typ für Aufträge mit AWS Glue-Version 2.0 oder höher • **`G.2X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker wird 2 zugeordnet DPUs (8 VCPUs, 32 GB Arbeitsspeicher, 128 GB Festplatte) und stellt einen Executor pro Worker bereit. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge und solche, die Machine-Learning-Transformationen ausführen.
Logs (Protokolle)	Ein Link zu den Auftragsprotokollen für die kontinuierliche Protokollierung (`/aws-glue/jobs/logs-v2`).
Output Logs (Ausgabeprotokolle)	Ein Link zu den Auftrags-Ausgabeprotokolldateien (`/aws-glue/jobs/output`).
Fehlerprotokolle	Ein Link zu den Auftrags-Fehlerprotokolldateien (`/aws-glue/jobs/error`).

Sie können auch die folgenden zusätzlichen Elemente anzeigen, die verfügbar sind, wenn Sie Informationen zu den letzten Auftragsausführungen aufrufen. Weitere Informationen finden Sie unter the section called “Anzeigen von Informationen zu den letzten Auftragsausführungen”.

- Eingabeargumente
- Kontinuierliche Protokolle
- Metriken – Sie können sich Visualisierungen einfacher Metriken ansehen. Weitere Informationen zu den enthaltenen Metriken finden Sie unter the section called “Amazon CloudWatch Metriken für eine Spark-Jobausführung anzeigen”.
- Spark-UI: Sie können Spark-Protokolle für einen Auftrag in der Spark-Benutzeroberfläche visualisieren. Weitere Informationen zur Verwendung der Spark-Web-UI finden Sie unter the section called “Überwachen über die Spark-Benutzeroberfläche”. Aktivieren Sie dieses Feature, indem Sie den Schritten in the section called “Aktivieren der Spark-Benutzeroberfläche für Aufgaben” folgen.

Amazon CloudWatch Metriken für eine Spark-Jobausführung anzeigen

Auf der Detailseite für einen Joblauf, unterhalb des Abschnitts Ausführungsdetails, können Sie die Job-Metriken einsehen. AWS Glue Studiosendet Job-Metriken Amazon CloudWatch für jeden Job-Lauf an.

AWS Glue meldet Amazon CloudWatch alle 30 Sekunden Metriken. Die AWS Glue -Metriken stellen Deltawerte gegenüber den zuvor gemeldeten Werten dar. Gegebenenfalls aggregieren (summieren) die Metrik-Dashboards die 30-Sekunden-Wertezu einem Wert für die gesamte vergangene Minute. Bei den Apache Spark-Metriken, AWS Glue die an weitergegeben Amazon CloudWatch werden, handelt es sich jedoch im Allgemeinen um absolute Werte, die den aktuellen Status zum Zeitpunkt der Berichterstattung darstellen.

> **Note**
>
> Sie müssen Ihr Konto für den Zugriff auf Amazon CloudWatch, konfigurieren.

Die Metriken enthalten Informationen über die Auftragsausführung, z. B.:

- ETL data movement (ETL-Datenverschiebung) – die Anzahl der Bytes, die von Amazon S3 gelesen oder darin geschrieben wurden.

- Memory profile: Heap used (Speicherprofil: verwendeter Heap) – die Anzahl der vom Heap der Java Virtual Machine (JVM) verwendeten Speicherbytes.
- Memory profile: Heap usage (Speicherprofil: Heap-Nutzung) – der prozentuale Anteil des vom JVM-Heap verwendeten Speichers (zwischen 0 und 1).
- CPU load (CPU-Last) – der verwendete prozentuale Anteil der CPU-Systemauslastung (zwischen 0 und 1).

Amazon CloudWatch Metriken für einen Ray-Joblauf anzeigen

Auf der Detailseite für einen Joblauf, unterhalb des Abschnitts Ausführungsdetails, können Sie die Job-Metriken einsehen. AWS Glue Studiosendet Job-Metriken Amazon CloudWatch für jeden Job-Lauf an.

AWS Glue meldet Amazon CloudWatch alle 30 Sekunden Metriken. Die AWS Glue -Metriken stellen Deltawerte gegenüber den zuvor gemeldeten Werten dar. Gegebenenfalls aggregieren (summieren) die Metrik-Dashboards die 30-Sekunden-Wertezu einem Wert für die gesamte vergangene Minute. Bei den Apache Spark-Metriken, AWS Glue die an weitergegeben Amazon CloudWatch werden, handelt es sich jedoch im Allgemeinen um absolute Werte, die den aktuellen Status zum Zeitpunkt der Berichterstattung darstellen.

> **Note**
>
> Sie müssen Ihr Konto für den Zugriff konfigurieren Amazon CloudWatch, wie unter beschrieben.

In Ray-Aufträgen können Sie die folgenden aggregierten Metrikdiagramme anzeigen. Damit können Sie ein Profil Ihres Clusters und Ihrer Aufgaben entwickeln und auf detaillierte Informationen über jeden Knoten zugreifen. Die Zeitreihendaten, die diesen Diagrammen zugrunde liegen, stehen CloudWatch für weitere Analysen zur Verfügung.

Aufgabenprofil: Status der Aufgabe

Zeigt die Anzahl der Ray-Aufgaben im System an. Jedem Aufgabenlebenszyklus wird eine eigene Zeitreihe zugewiesen.

Aufgabenprofil: Name der Aufgabe

Zeigt die Anzahl der Ray-Aufgaben im System an. Es werden nur ausstehende und aktive Aufgaben angezeigt. Jedem Aufgabentyp (nach Namen) wird eine eigene Zeitreihe zugewiesen.

Cluster-Profil: wird CPUs verwendet

Zeigt die Anzahl der verwendeten CPU-Kerne an. Jedem Knoten wird eine eigene Zeitreihe zugewiesen. Knoten werden durch IP-Adressen identifiziert, die kurzlebig sind und nur zur Identifizierung verwendet werden.

Cluster-Profil: Speichernutzung des Objektspeichers

Zeigt die Speichernutzung durch den Ray-Objekt-Cache an. Jedem Speicherort (physischer Speicher, auf der Festplatte zwischengespeichert und in Amazon S3 verschüttet) wird eine eigene Zeitreihe zugewiesen. Der Objektspeicher verwaltet die Datenspeicherung auf allen Knoten im Cluster. Weitere Informationen finden Sie unter Objekte in der Ray-Dokumentation.

Cluster-Profil: Anzahl der Knoten

Zeigt die Anzahl der für den Cluster bereitgestellten Knoten an.

Knotendetail: CPU-Auslastung

Zeigt die CPU-Auslastung auf jedem Knoten als Prozentsatz an. Jede Reihe zeigt einen aggregierten Prozentsatz der CPU-Auslastung aller Kerne auf dem Knoten.

Knotendetail: Speichernutzung

Zeigt die Speichernutzung auf jedem Knoten in GB an. Jede Reihe zeigt den zwischen allen Prozessen auf dem Knoten zusammengefassten Speicher, einschließlich Ray-Aufgaben und dem Plasma-Speicherprozess. Dies gilt nicht für auf der Festplatte gespeicherte oder an Amazon S3 übertragene Objekte.

Knotendetail: Festplattennutzung

Zeigt die Festplattennutzung auf jedem Knoten in GB an.

Knotendetail: I/O Festplattengeschwindigkeit

Zeigt die Festplatte I/O auf jedem Knoten in KB/s an.

Knotendetail: Netzwerkdurchsatz I/O

Zeigt das Netzwerk I/O auf jedem Knoten in KB/s an.

Knotendetail: CPU-Auslastung durch Ray-Komponente

Zeigt die CPU-Nutzung in Teilkernen an. Jeder Ray-Komponente an jedem Knoten wird eine eigene Zeitreihe zugewiesen.

Knotendetail: Speichernutzung durch Ray-Komponente

Zeigt die Speichernutzung in GB an. Jeder Ray-Komponente an jedem Knoten wird eine eigene Zeitreihe zugewiesen.

Erkennen und Verarbeiten von sensiblen Daten

Die Detect PII-Transformation identifiziert persönlich identifizierbare Informationen (PII) in Ihrer Datenquelle. Sie wählen die PII-Entität aus, um zu identifizieren, wie die Daten gescannt werden sollen und was mit der PII-Entität zu tun ist, die durch die Detect PII-Transformation identifiziert wurde.

Mit der Detect PII-Transformation lassen sich Entitäten erkennen, maskieren oder entfernen, die von Ihnen oder AWS definiert werden. Dies steigert die Compliance und senkt Haftungsrisiken. Beispielsweise möchten Sie möglicherweise sicherstellen, dass Ihre Daten keine personenbezogenen Daten enthalten, die gelesen werden können, und Sie möchten Sozialversicherungsnummern mit einer festen Zeichenfolge (z. B. xxx-xx-xxxx), Telefonnummern oder Adressen maskieren.

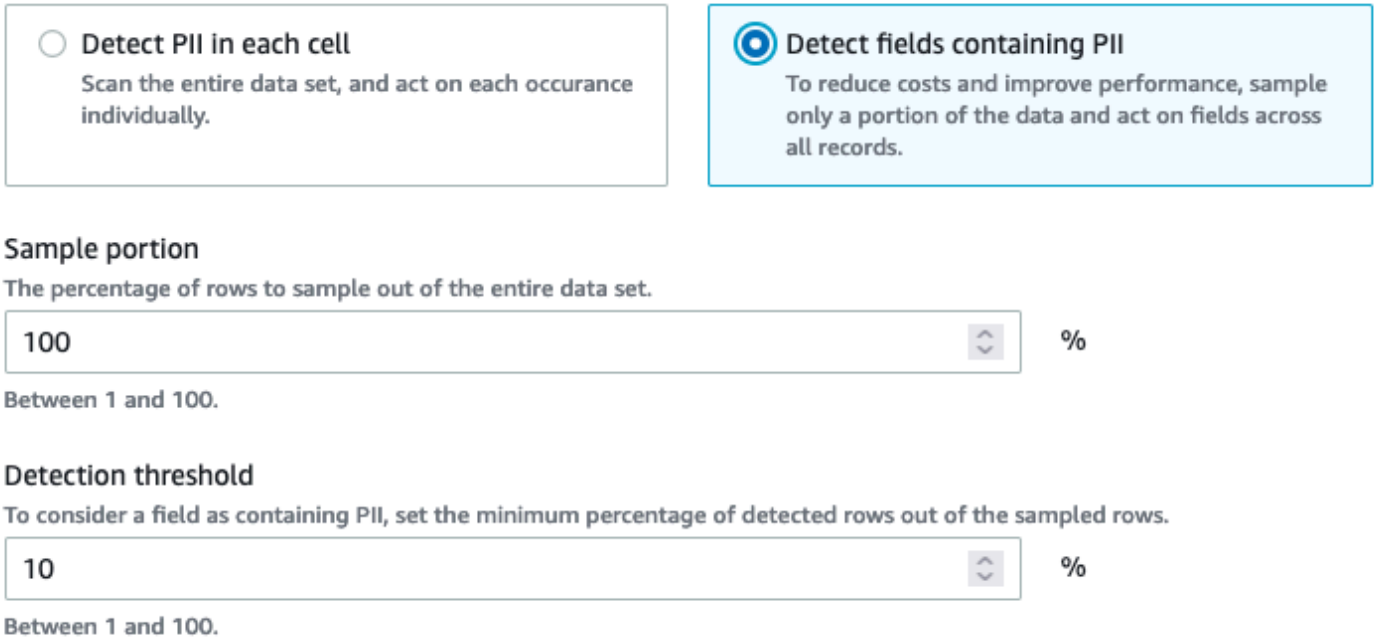
Informationen zum Arbeiten mit sensiblen Daten außerhalb von AWS Glue Studio finden Sie unter Erkennung sensibler Daten außerhalb von AWS Glue Studio verwenden.

Themen

- Auswahl der Scan-Methode der Daten
- Auswählen der zu erkennenden PII-Entitäten
- Angeben der Erkennungsempfindlichkeit
- Auswahl, was mit identifizierten PII-Daten zu tun ist
- Hinzufügen detaillierter Aktionsüberschreibungen

Auswahl der Scan-Methode der Daten

Wenn Sie Ihren Datensatz nach sensiblen Daten wie persönlich identifizierbaren Informationen (PII) durchsuchen, haben Sie die Wahl, PII in jeder Zeile zu erkennen oder die Spalten zu erkennen, die PII-Daten enthalten.



Wenn Sie Detect PII in each cell (PII in jeder Zelle erkennen) wählen, entscheiden Sie sich für das Scannen aller Zeilen in der Datenquelle. Dies ist ein umfassender Scan, um sicherzustellen, dass PII-Entitäten identifiziert werden.

Wenn Sie Detect fields containing PII (Felder mit PII erkennen) wählen, entscheiden Sie sich für das Scannen von Stichproben von Reihen auf PII-Entitäten. Dies ist eine Möglichkeit, Kosten und Ressourcen unten zu halten und gleichzeitig die Felder zu identifizieren, in denen PII-Entitäten gefunden werden.

Wenn Sie sich dafür entscheiden, Felder zu erkennen, die PII enthalten, können Sie Kosten reduzieren und Leistung durch die Anwendung von Stichprobenverfahren auf eine Teilmenge von Zeilen verbessern. Wenn Sie diese Option auswählen, können Sie zusätzliche Optionen angeben:

- Sample portion (Stichproben-Teilmenge): Auf diese Weise können Sie den Prozentsatz der Zeile für die Stichprobe angeben. Wenn Sie beispielsweise „50“ eingeben, geben Sie an, dass Sie 50 Prozent der gescannten Zeilen für die PII-Entität wünschen.

- Detection threshold (Schwellenwert der Erkennung): Auf diese Weise können Sie den Prozentsatz der Zeilen angeben, welche die PII-Entität enthalten, damit die gesamte Spalte als PII-Entität identifiziert wird. Wenn Sie beispielsweise „10“ eingeben, geben Sie an, dass die Nummer der PII-Entität, US Phone, in den gescannten Zeilen mindestens 10 Prozent betragen muss, damit das Feld als PII-Entität, US Phone, identifiziert wird. Wenn der Prozentsatz der Zeilen, welche die PII-Entität enthalten, weniger als 10 Prozent beträgt, wird dieses Feld nicht als PII-Entität, US Phone, bezeichnet.

Auswählen der zu erkennenden PII-Entitäten

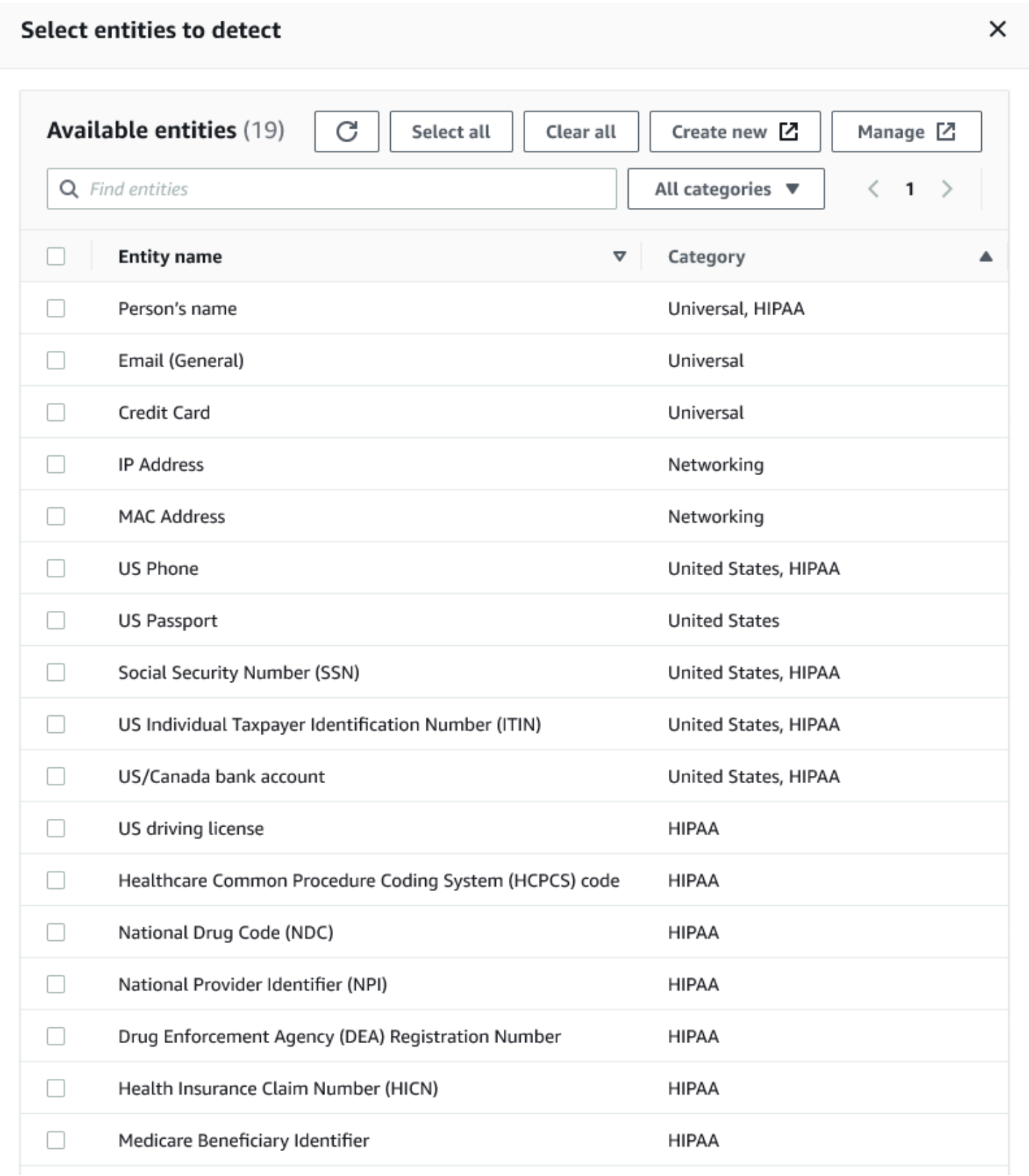
Wenn Sie Detect PII in each cell (PII in jeder Zelle erkennen) wählen, haben Sie drei Optionen:

- Alle verfügbaren PII-Muster — dazu gehören auch AWS Entitäten.
- Kategorien auswählen – Wenn Sie Kategorien auswählen, enthalten PII-Muster automatisch Muster in den von Ihnen ausgewählten Kategorien.
- „Select specific patterns“ (Bestimmte Muster auswählen) – Nur die ausgewählten Muster werden erkannt.

Eine vollständige Liste der verwalteten vertraulichen Datentypen finden Sie unter Verwaltete Datentypen.

Auswählen aus allen verfügbaren PII-Mustern

Wenn Sie Alle verfügbaren PII-Muster wählen, wählen Sie Entitäten aus, die von vordefiniert sind. AWS Sie können eine, mehrere oder alle Entitäten auswählen.


Select entities to detect
Available entities (19)
Select all
Clear all
Create new
Manage
Find entities
All categories
1
Entity name
Category
Person's name
Universal, HIPAA
Email (General)
Universal
Credit Card
Universal
IP Address
Networking
MAC Address
Networking
US Phone
United States, HIPAA
US Passport
United States
Social Security Number (SSN)
United States, HIPAA
US Individual Taxpayer Identification Number (ITIN)
United States, HIPAA
US/Canada bank account
United States, HIPAA
US driving license
HIPAA
Healthcare Common Procedure Coding System (HCPCS) code
HIPAA
National Drug Code (NDC)
HIPAA
National Provider Identifier (NPI)
HIPAA
Drug Enforcement Agency (DEA) Registration Number
HIPAA
Health Insurance Claim Number (HICN)
HIPAA
Medicare Beneficiary Identifier
HIPAA

Kategorien auswählen

Wenn Sie Select categories (Kategorien auswählen) zum Erkennen von PII-Mustern ausgewählt haben, können Sie aus den Optionen im Dropdown-Menü auswählen. Beachten Sie, dass einige Entitäten mehreren Kategorien angehören können. Zum Beispiel fällt die Entität Person's name (Name der Person) in die Kategorien Universal (Universell) und HIPAA.

- „Universal“ (Universell), z. B. „Email“ (E-Mail), „Credit Card“ (Kreditkarte)
- HIPAA (Beispiele: US-Führerschein, Healthcare Common Procedure Coding System (HCPCS)-Code)
- „Networking“ (Netzwerk), z. B. „IP-Address“ (IP-Adresse), „MAC-Address“ (MAC-Adresse)
- Argentinien
- Australien
- Österreich
- Belgien
- Bosnien
- Bulgarien
- Kanada
- Chile
- Kolumbien
- Kroatien
- Zypern
- Tschechien
- Dänemark
- Estland
- Finnland
- Frankreich
- Deutschland
- Griechenland
- Ungarn
- Irland
- Korea

- Japan
- Mexiko
- Niederlande
- Neuseeland
- Norwegen
- Portugal
- Rumänien
- Singapur
- Slowakei
- Slowenien
- Spanien
- Schweden
- Schweiz
- Türkei
- Ukraine
- Vereinigte Staaten
- Großbritannien und Nordirland
- Venezuela

Bestimmte Muster auswählen

Wenn Sie Select specific patterns (Bestimmte Muster auswählen) zum Erkennen von PII-Mustern verwenden, können Sie eine Liste von bereits erstellten Mustern durchsuchen oder ein neues Muster zur Erkennung von Entitäten erstellen.

In den folgenden Schritten wird beschrieben, wie Sie ein neues benutzerdefiniertes Muster zum Erkennen sensibler Daten erstellen. Sie erstellen das benutzerdefinierte Muster, indem Sie einen Namen für das benutzerdefinierte Muster eingeben, einen regulären Ausdruck hinzufügen und optional Kontextwörter definieren.

1. Um ein neues Muster zu erstellen, klicken Sie auf Create new (Neues erstellen).

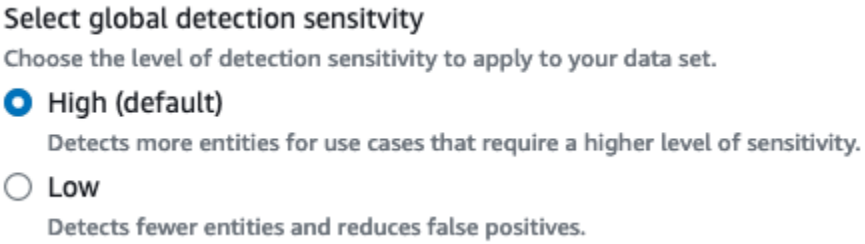
2. Geben Sie auf der Seite „Create detection entity“ (Entität zur Erkennung erstellen) den Entitätsnamen und einen regulären Ausdruck ein. Der reguläre Ausdruck (Regex) wird von AWS Glue verwendet, um Entitäten abzugleichen.
3. Klicken Sie auf Validate (Validieren). Wenn die Validierung erfolgreich ist, wird eine Bestätigungsmeldung angezeigt, die besagt, dass die Zeichenfolge ein gültiger regulärer Ausdruck ist. Wenn die Validierung nicht erfolgreich ist, wird eine Meldung angezeigt, die besagt, dass die Zeichenfolge nicht der richtigen Formatierung und den akzeptierten Zeichenliteralen, Operatoren oder Konstrukten entspricht.
4. Sie können zusätzlich zum regulären Ausdruck Kontextwörter hinzufügen. Kontextwörter können die Wahrscheinlichkeit einer Übereinstimmung erhöhen. Sie können in Fällen nützlich sein, in denen Feldnamen die Entität nicht beschreiben. Beispielsweise können US-Sozialversicherungsnummern (Social Security Numbers) „SSN“ oder „SS“ genannt werden. Das Hinzufügen dieser Kontextwörter kann helfen, die Entität abzugleichen.
5. Klicken Sie auf Create (Erstellen), um eine Entität zur Erkennung zu erstellen. Erstellte Entitäten werden in der AWS Glue Studio-Konsole angezeigt. Klicken Sie Detection entities (Erkennungsentitäten) im linken Navigationsmenü.

   Sie können Entitäten zur Erkennung auf der Seite Detection entities (Erkennungsentitäten) bearbeiten, löschen oder erstellen. Sie können auch über das Suchfeld nach einem Muster suchen.

Angeben der Erkennungsempfindlichkeit

Sie können für die Erkennung sensibler Daten den Grad der Empfindlichkeit festlegen.

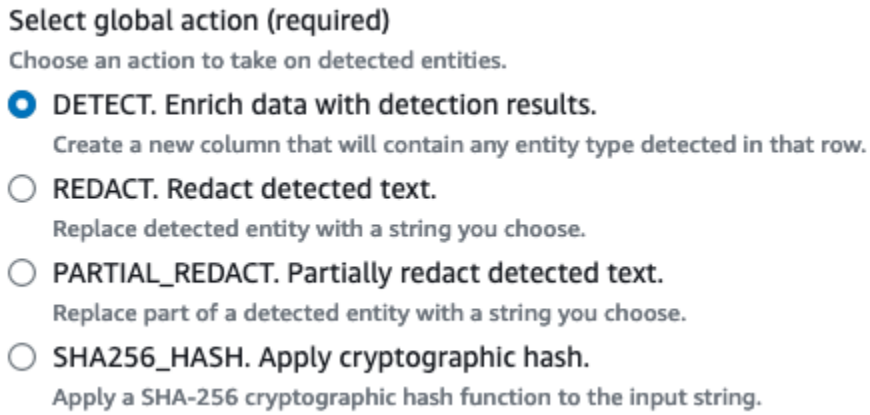
- Hoch – (Standard) Erkennt mehr Entitäten für Anwendungsfälle, die einen höheren Empfindlichkeitsgrad erfordern. Für alle AWS Glue-Aufträge, die nach November 2023 erstellt wurden, ist diese Einstellung automatisch aktiviert.
- Niedrig – Erkennt weniger Entitäten und reduziert Fehlalarme.



Auswahl, was mit identifizierten PII-Daten zu tun ist

Wenn Sie PII in der gesamten Datenquelle erkennen möchten, können Sie eine globale Aktion auswählen:

- Daten mit Erkennungsergebnissen bereichern: Wenn Sie in jeder Zelle Detect PII ausgewählt haben, können Sie die erkannten Entitäten in einer neuen Spalte speichern.
- Redigieren von erkanntem Text: Sie können den erkannten PII-Wert durch eine Zeichenfolge ersetzen, die Sie im optionalen Texteingabefeld „Ersetzen“ angeben. Wenn keine Zeichenfolge angegeben wird, wird die erkannte PII-Entität durch ,******* , ersetzt.
- Erkannten Text redigieren: Sie können den erkannten PII-Wert durch eine Zeichenfolge Ihrer Wahl ersetzen. Sie haben zwei Optionen. Entweder Sie lassen die Enden unmaskiert oder Sie geben ein explizites Regex-Muster zur Maskierung an. Diese Funktion ist in AWS Glue 2.0 nicht verfügbar.
- Apply cryptographic hash: (Anwendung eines kryptografischen Hashes): Sie können den erkannten PII-Wert an eine kryptografische SHA-256-Hash-Funktion übergeben und den Wert durch die Ausgabe der Funktion ersetzen.



Unterschiede zwischen den AWS Glue-Versionen 2.0 und 3.0+

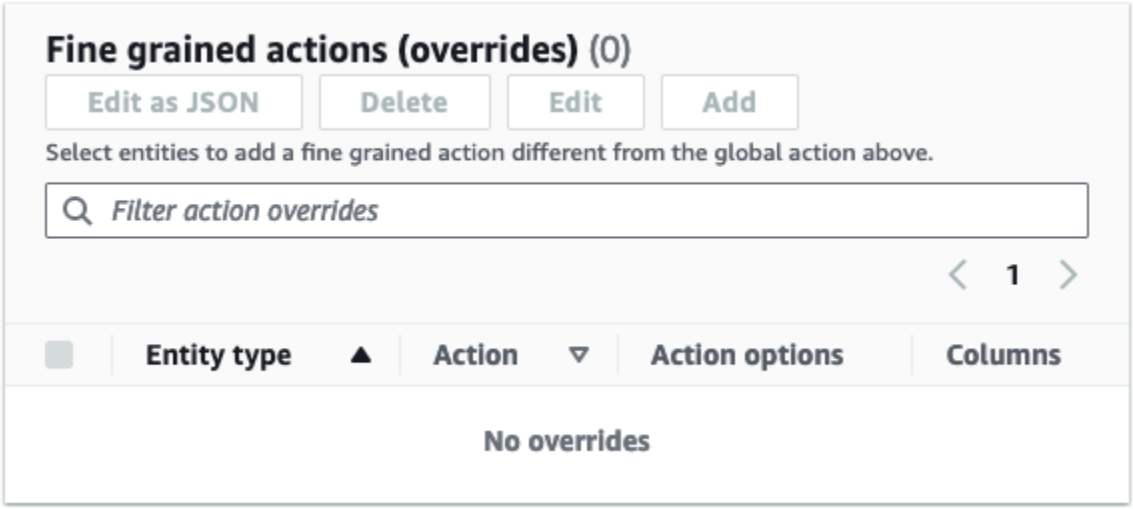
AWS Glue2.0-Jobs geben für jede Spalte in einer zusätzlichen Spalte eine neue Nachricht DataFrame mit den erkannten PII-Informationen zurück. Jede Redigierung oder Hash-Bearbeitung ist innerhalb des AWS Glue-Skripts auf der visuellen Registerkarte sichtbar.

AWS GlueJobs der Typen 3.0 und 4.0 geben einen neuen Wert DataFrame mit derselben zusätzlichen Spalte zurück. Ein neuer Schlüssel für „actionUsed“ ist vorhanden und kann einen der folgenden Werte haben: `DETECT`, `REDACT`, `PARTIAL_REDACT` oder `SHA256_HASH`. Wenn eine Maskierungsaktion ausgewählt ist, DataFrame werden Daten zurückgegeben, bei denen sensible Daten maskiert sind.

Hinzufügen detaillierter Aktionsüberschreibungen

Zusätzliche Erkennungs- und Aktionseinstellungen können der Tabelle mit detaillierten Aktionsüberschreibungen hinzugefügt werden. Das ermöglicht Ihnen Folgendes:

- Bestimmte Spalten für die Erkennung einschließen oder ausschließen: Ein abgeleitetes Schema für die Datenquelle füllt die Tabelle mit verfügbaren Spalten.
- Einstellungen angeben, die detaillierter sind als globale Aktionen: Sie können beispielsweise unterschiedliche Einstellungen für die Textredigierung für verschiedene Entitätstypen angeben.
- Eine andere Aktion als die globale Aktion angeben: Wenn Sie eine andere Aktion für einen anderen sensiblen Datentyp anwenden möchten, können Sie das hier tun. Beachten Sie, dass zwei verschiedene edit-in-place Aktionen (Schwärzen und Hashing) nicht für dieselbe Spalte verwendet werden können, Detect jedoch immer verwendet werden kann.



Verwaltung von ETL-Jobs mit AWS Glue Studio

Sie können die einfache grafische Oberfläche verwenden in AWS Glue Studio um Ihre ETL-Jobs zu verwalten. Wählen Sie im Navigationsmenü Jobs (Aufträge) aus. Es öffnet sich die Seite Jobs (Aufträge). Auf dieser Seite können Sie alle Jobs sehen, die Sie entweder mit erstellt haben AWS Glue Studio oder der AWS Glue console. Auf dieser Seite können Sie Ihre Aufträge ansehen, verwalten und ausführen.

Hier sind außerdem folgende Aktionen möglich:

- Starten einer Auftragsausführung
- Planen von Auftragsausführungen
- Verwalten von Auftragsplänen
- Anhalten von Auftragsausführungen
- Anzeigen von Aufträgen
- Anzeigen von Informationen zu den letzten Auftragsausführungen
- Anzeigen des Auftragsskripts
- Ändern der Auftragseigenschaften
- Speichern des Auftrags
- Klonen von Aufträgen
- Löschen von Aufträgen

Starten einer Auftragsausführung

In AWS Glue Studio, Sie können Ihre Jobs bei Bedarf ausführen. Ein Job kann mehrfach ausgeführt werden, und jedes Mal, wenn Sie den Job ausführen, AWS Glue sammelt Informationen über die Jobaktivitäten und die Leistung. Diese Informationen werden unter dem Begriff der Auftragsausführung zusammengefasst, die eine eigene Auftragsausführungs-ID erhält.

Sie können eine Auftragsausführung auf folgende Weise initiieren AWS Glue Studio:

- Wählen Sie auf der Seite Jobs (Aufträge) den Auftrag aus, den Sie starten möchten. Wählen Sie dann die Schaltfläche Run job (Auftrag ausführen) aus.
- Wenn Sie einen Auftrag im visuellen Editor ansehen und der Auftrag gespeichert wurde, können Sie das Kontrollkästchen Run (Ausführen) auswählen, um eine Auftragsausführung zu starten.

Weitere Informationen zu Auftragsausführungen finden Sie unter Arbeiten mit Jobs auf der AWS Glue Konsole im AWS Glue Entwicklerhandbuch.

Planen von Auftragsausführungen

In AWS Glue Studio, Sie können einen Zeitplan erstellen, nach dem Ihre Jobs zu bestimmten Zeiten ausgeführt werden. Sie können verschiedene Einstellungen treffen, z. B. wie oft oder an welchen Wochentagen und zu welcher Uhrzeit Aufträge oder Crawler ausgeführt werden sollen. Diese Einstellungen beruhen auf `cron`, wobei auch dieselben Einschränkungen wie für `cron` gelten. Wenn Sie z. B. Ihren Auftrag immer am 31. des Monats ausführen möchten, denken Sie daran, dass einige Monate keine 31 Tage haben. Weitere Informationen über `cron` finden Sie unter Cron-Ausdrücke im AWS Glue -Entwicklerhandbuch.

Aufträge nach einem Zeitplan ausführen

1. Erstellen Sie einen Auftragszeitplan mithilfe einer der folgenden Methoden:

   - Wählen Sie auf der Seite Jobs (Aufträge) den Auftrag aus, für den Sie einen Zeitplan erstellen möchten. Wählen Sie dann Actions (Aktionen) und danach Schedule job (Auftrag planen) aus.
   - Wenn Sie einen Auftrag im visuellen Editor ansehen und der Auftrag gespeichert wurde, wählen Sie die Registerkarte Schedules (Zeitpläne) aus. Wählen Sie dann Create Schedule (Zeitplan erstellen) aus.

2. Geben Sie auf der Seite Schedule job run (Auftragsausführung planen) die folgenden Angaben ein:

   - Name: Geben Sie einen Namen für Ihren Auftragszeitplan ein.
   - Frequency (Häufigkeit): Geben Sie an, wie oft der Auftrag ausgeführt werden soll. Es gibt die folgenden Optionen:
     - Hourly (Stündlich): Der Auftrag wird einmal jede Stunde zu einer bestimmten Minute ausgeführt. Sie können die Minute bestimmten, in der der Auftrag ausgeführt werden soll. Standardmäßig wird der Auftrag bei der stündlichen Ausführung am Anfang der Stunde ausgeführt (Minute 0).
     - Daily (Täglich): Der Auftrag wird einmal am Tag zu einem festgelegten Zeitpunkt ausgeführt. Sie können bestimmen, um welche Minute und zu welcher Start hour (Stunde zum Starten) der Auftrag ausgeführt werden soll. Die Stunden werden im 24-Stunden-Format (0 bis 23 Uhr) angegeben. Die Standardwerte für Minute und Stunde sind 0, was bedeutet, dass

bei der täglichen (daily) Ausführung der Auftrag standardmäßig um Mitternacht ausgeführt wird.

- Weekly (Wöchentlich): Der Auftrag wird jede Woche an einem oder mehreren Tagen ausgeführt. Zusätzlich zu den zuvor beschriebenen Einstellungen für „Daily (Täglich)“ können Sie auch Wochentage auswählen, an denen der Auftrag ausgeführt werden soll. Sie können einen oder mehrere Tage auswählen.
- Monthly (Monatlich): Der Auftrag wird jeden Monat an einem bestimmten Tag ausgeführt. Zusätzlich zu den zuvor beschriebenen Einstellungen für „Daily (Täglich)“ können Sie die auch den Tag im Monat auswählen, an dem der Auftrag ausgeführt werden soll. Geben Sie den Tag als numerischen Wert von 1 bis 31 an. Wenn Sie einen Tag auswählen, der in einem bestimmten Monat nicht vorkommt, z. B. den 30., dann wird der Auftrag, im Februar nicht ausgeführt.
- Custom (Benutzerdefiniert): Geben Sie einen Ausdruck für Ihren Zeitplan mithilfe der `cron`-Syntax an. Cron-Ausdrücke ermöglichen es Ihnen, komplexere Zeitpläne zu erstellen, bei denen der Auftrag anstelle eines bestimmten Tags im Monat zum Beispiel am letzten Tag des Monats ausgeführt wird oder jeden dritten Monat am 7. und am 21.

  Siehe Cron-Ausdrücke im AWS Glue -Entwicklerhandbuch

- Description (Beschreibung): Sie können optional eine Beschreibung für Ihren Auftragszeitplan eingeben. Wenn Sie denselben Zeitplan für mehrere Aufträge verwenden möchten, lässt sich anhand einer Beschreibung leichter erkennen, wann ein Auftragszeitplan die Ausführung vorsieht.

3. Klicken Sie auf Create schedule (Zeitplan erstellen), um den Auftragszeitplan zu speichern.
4. Nachdem Sie den Zeitplan erstellt haben, erscheint oben auf der Konsolenseite eine Bestätigungsmeldung. In dem Banner sehen Sie unter Job details (Auftragsdetails) Einzelheiten zum Auftrag. Dadurch öffnet sich die Seite des visuellen Auftragseditors mit der aktiven Registerkarte Schedules (Zeitpläne).

Verwalten von Auftragsplänen

Nachdem Sie Zeitpläne für einen Auftrag erstellt haben, können Sie den Auftrag im visuellen Editor öffnen und mit der Registerkarte Schedules (Zeitpläne) die Zeitpläne verwalten.

Auf der Registerkarte Schedules (Zeitpläne) im visuellen Editor können Sie die folgenden Aufgaben ausführen:

- Erstellen eines neuen Zeitplans

  Wählen Sie Create schedule (Zeitplan erstellen) aus und geben Sie dann die Informationen für Ihren Zeitplan ein, wie unter the section called “Planen von Auftragsausführungen” beschrieben.

- Bearbeiten eines bestehenden Zeitplans.

  Wählen Sie den Zeitplan aus, den Sie bearbeiten möchten, und dann Action (Aktion) und anschließend Edit schedule (Zeitplan bearbeiten). Wenn Sie einen vorhandenen Zeitplan bearbeiten möchten, wird die Frequency (Häufigkeit) als Custom (Benutzerdefiniert) angezeigt und der Zeitplan als `cron`-Ausdruck. Sie können entweder den `cron`-Ausdruck ändern oder einen neuen Zeitplan mit der Schaltfläche Frequency (Häufigkeit) angeben. Sobald Sie Ihre Änderungen vorgenommen haben, wählen Sie Update schedule (Zeitplan ändern) aus.

- Pausieren eines aktiven Zeitplans.

  Wählen Sie einen aktiven Zeitplan und dann Action (Aktion) aus, gefolgt von Pause schedule (Zeitplan pausieren). Der Zeitplan wird daraufhin sofort deaktiviert. Wählen Sie die Schaltfläche zum Aktualisieren (refresh/reload) aus, um den geänderten Auftragsplanstatus anzuzeigen.

- Einen angehaltenen Zeitplan wieder starten.

  Wählen Sie einen aktiven Zeitplan und dann Action (Aktion) aus, gefolgt von Resume schedule (Zeitplan wieder starten). Der Zeitplan wird daraufhin sofort aktiviert. Wählen Sie die Schaltfläche zum Aktualisieren (refresh/reload) aus, um den geänderten Auftragsplanstatus anzuzeigen.

- Löschen eines Zeitplans.

  Wählen Sie den Zeitplan aus, den Sie löschen möchten, und dann Action (Aktion) und anschließend Delete schedule (Zeitplan löschen). Der Zeitplan wird daraufhin sofort gelöscht. Wählen Sie die Schaltfläche zum Aktualisieren (refresh/reload) aus, um die geänderte Auftragsplanliste anzuzeigen. Der Zeitplan zeigt den Status Deleting (Wird gelöscht) an, bis er vollständig entfernt wurde.

Anhalten von Auftragsausführungen

Sie können einen Auftrag anhalten, bevor die Auftragsausführung abgeschlossen ist. Diese Option können Sie wählen, wenn Sie wissen, dass der Auftrag nicht korrekt konfiguriert ist oder er zu lange dauert.

Gehen Sie auf der Seite Monitoring (Überwachung) zur Liste Job runs (Auftragsausführungen) und wählen Sie dort den Auftrag aus, den Sie anhalten möchten. Wählen Sie Actions (Aktionen) und danach Stop run (Ausführung anhalten) aus.

Anzeigen von Aufträgen

Sie können alle Ihre Aufträge auf der Seite Jobs (Aufträge) sehen. Auf diese Seite gelangen Sie über Jobs (Aufträge) im Navigationsbereich.

Auf der Seite Jobs (Aufräge) können Sie alle Aufträge sehen, die in Ihrem Konto erstellt wurden. Unter Your jobs (Ihre Aufträge) sind die Aufträge nach Name, Typ, Status der letzten Ausführung sowie Erstellungsdatum und Datum der letzten Änderung aufgelistet. Sie können den Namen eines Auftrags auswählen, um detaillierte Informationen für diesen Auftrag zu sehen.

Sie können auch über das Überwachungs-Dashboard alle Ihre Aufträge sehen. Zum Dashboard gelangen Sie über Monitoring (Überwachung) im Navigationsbereich.

Anpassen der Auftragsanzeige

Sie können anpassen, wie die Aufträge im Abschnitt Your jobs (Ihre Aufträge) auf der Seite Jobs (Aufträge) angezeigt werden. Außerdem können Sie Text in das Suchtextfeld eingeben, sodass nur Aufträge mit einem Namen erscheinen, der diesen Text enthält.

Wenn Sie

im Bereich „Ihre Jobs“ auf das Einstellungssymbol klicken, können Sie festlegen, wie AWS Glue Studio zeigt die Informationen in der Tabelle an. Sie können die Textzeilen in der Anzeige umbrechen, die Anzahl der auf der Seite angezeigten Aufträge ändern und einstellen, welche Spalten zu sehen sein sollen.

Anzeigen von Informationen zu den letzten Auftragsausführungen

Ein Auftrag kann mehrmals ausgeführt werden, wenn neue Daten am Quellspeicherort hinzugefügt werden. Jedes Mal, wenn ein Auftrag ausgeführt wird, wird der Auftragsausführung eine eindeutige ID zugewiesen und Informationen zu dieser Auftragsausführung gesammelt. Diese Informationen können Sie mithilfe der folgenden Methoden anzeigen:

- Wählen Sie die Registerkarte Runs (Ausführungen) des visuellen Editors, um die Informationen zur Auftragsausführung für den aktuell angezeigten Auftrag zu sehen.

Auf der Registerkarte Runs (Ausführungen) auf der Seite Recent job runs (Letzte Auftragsausführungen) gibt es für jede Auftragsausführung eine Karte. Die Informationen auf der Registerkarte Runs (Ausführungen) umfassen Folgendes:

- Auftragsausführungs-ID
- Anzahl der Ausführungversuche dieses Auftrags
- Status der Auftragsausführung
- Start- und Endzeit für die Auftragsausführung
- Laufzeit für die Auftragsausführung
- Link zu den Auftrags-Protokolldateien
- Link zu den Auftrags-Fehlerprotokolldateien
- Welcher Fehler bei fehlgeschlagenen Aufträgen ausgegeben wurde

- Sie können eine Auftragsausführung auswählen, um zusätzliche Informationen zum Auftrag anzuzeigen. Unter anderem:
  - Eingabeargumente
  - Kontinuierliche Protokolle
  - Metriken – Sie können sich Visualisierungen einfacher Metriken ansehen. Weitere Informationen zu den enthaltenen Metriken finden Sie unter the section called “Amazon CloudWatch Metriken für eine Spark-Jobausführung anzeigen”.
  - Spark-UI: Sie können Spark-Protokolle für einen Auftrag in der Spark-Benutzeroberfläche visualisieren. Weitere Informationen zur Verwendung der Spark-Web-UI finden Sie unter the section called “Überwachen über die Spark-Benutzeroberfläche”. Aktivieren Sie dieses Feature, indem Sie den Schritten in the section called “Aktivieren der Spark-Benutzeroberfläche für Aufgaben” folgen.

Sie können Details anzeigen auswählen, um ähnliche Informationen auf der Seite mit den Details zur Auftragsausführung anzuzeigen. Alternativ können Sie die Seite mit den Details zur Auftragsausführung über die Seite Überwachung aufrufen. Wählen Sie im Navigationsbereich Monitoring (Überwachung) aus. Scrollen Sie nach unten bis zur Liste Job runs (Auftragsausführungen). Wählen Sie den Auftrag aus und dann View run Details (Details zur Auftragsausführung anzeigen). Die Inhalte sind unter Anzeigen der Details einer Auftragsausführung beschrieben.

Weitere Informationen zu den Auftragsprotokollen finden Sie unter Anzeigen der Auftragsausführungsprotokolle.

Anzeigen des Auftragsskripts

Nachdem Sie Informationen für alle Knoten im Job angegeben haben, AWS Glue Studio generiert ein Skript, das vom Job verwendet wird, um die Daten aus der Quelle zu lesen, die Daten zu transformieren und die Daten in den Zielordner zu schreiben. Wenn Sie den Auftrag speichern, können Sie dieses Skript jederzeit anzeigen.

Das generierte Skript für Ihren Auftrag anzeigen

1. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
2. Auf der Seite Jobs (Aufträge) wählen Sie aus der Liste Your jobs (Ihre Aufträge) den Namen des Auftrags aus, den Sie sich ansehen möchten. Alternativ können Sie einen Auftrag in der Liste auswählen. Wählen Sie dazu das Menü Actions (Aktionen) und dann Edit jobs (Auftrag bearbeiten) aus.
3. Wählen Sie auf der Seite des visuellen Editors oben die Registerkarte Script (Skript), um das Auftragsskript anzuzeigen.

   Infos zum Bearbeiten des Auftragsskripts finden Sie unter AWS Glue Leitfaden zur Programmierung.

Ändern der Auftragseigenschaften

Die Knoten im Auftragsdiagramm definieren die Aktionen, die vom Auftrag ausgeführt werden; es gibt jedoch einige weitere Eigenschaften, die Sie für den Auftrag konfigurieren können. Diese Eigenschaften bestimmen die Umgebung, in der der Auftrag ausgeführt wird, die verwendeten Ressourcen, die Schwellenwerteinstellungen, die Sicherheitseinstellungen und vieles mehr.

Die Umgebung der Auftragsausführung anpassen

1. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
2. Auf der Seite Jobs (Aufträge) wählen Sie aus der Liste Your jobs (Ihre Aufträge) den Namen des Auftrags aus, den Sie sich ansehen möchten.
3. Wählen Sie auf der Seite des visuellen Editors die Registerkarte Job details (Auftragsdetails) aus, die sich oben im Bereich zur Auftragsbearbeitung befindet.

4. Ändern Sie die Auftragseigenschaften wie gewünscht.

   Weitere Informationen zu den Auftragseigenschaften finden Sie unter Definieren von Auftragseigenschaften im AWS Glue -Entwicklerhandbuch.

5. Klappen Sie den Abschnitt Advanced properties (Erweiterte Eigenschaften) aus, wenn Sie diese zusätzlichen Auftragseigenschaften angeben möchten:

   - Script filename (Dateiname des Skripts) – Der Name der Datei, in der das Auftragsskript in Amazon S3 gespeichert wird.
   - Script path (Skript-Pfad) – Der Amazon-S3-Speicherort, an dem das Auftragsskript gespeichert wird.
   - Job-Metriken — (Nicht verfügbar für Python-Shell-Jobs) Aktiviert die Erstellung von Amazon CloudWatch Metriken, wenn dieser Job ausgeführt wird.
   - Kontinuierliche Protokollierung — (Nicht verfügbar für Python-Shell-Jobs) Aktiviert die kontinuierliche Protokollierung CloudWatch, sodass die Protokolle angezeigt werden können, bevor der Job abgeschlossen ist.
   - Spark UI und Spark UI logs path (Pfad für Spark-UI-Protokolle) – Aktiviert die Verwendung der Spark-Benutzeroberfläche für die Überwachung dieses Auftrags und gibt den Speicherort für die Spark-UI-Protokolle an (nicht verfügbar für Python-Shell-Aufträge).
   - Maximum concurrency (Max. Nebenläufigkeit) – legt fest, wie viele Ausführungen bei dieser Ausführung gleichzeitig stattfinden dürfen.
   - Temporärer Pfad — Der Speicherort eines Arbeitsverzeichnisses in Amazon S3, in das temporäre Zwischenergebnisse geschrieben werden, wenn AWS Glue führt das Job-Skript aus.
   - Delay notification threshold (minutes) (Schwellenwert für die Verzögerungsbenachrichtigung in Minuten) – gibt einen Verzögerungsschwellwert für den Auftrag an. Wenn der Job länger als im Schwellenwert angegeben ausgeführt wird, AWS Glue sendet eine Verzögerungsbenachrichtigung für den Job an CloudWatch.
   - Security configuration (Sicherheitskonfiguration) und Server-side encryption (Serverseitige Verschlüsselung) – zum Auswählen von Verschlüsselungsoptionen für den Auftrag.
   - Verwenden Sie den Glue-Datenkatalog als Hive-Metastore — Wählen Sie diese Option, wenn Sie den AWS Glue Data Catalog als Alternative zu Apache Hive Metastore.
   - Additional network connection (Zusätzliche Netzwerkverbindung) – für eine Datenquelle in einer VPC können Sie eine Verbindung vom Typ `Network` angeben, um sicherzustellen, dass der Auftrag über die VPC auf Ihre Daten zugreift.

- Python library path (Python-Bibliothekspfad), Dependent jars path (Abhängiger Jars-Pfad) (nicht für Python-Shell-Aufträge) oder Referenced files path (Pfad für referenzierte Dateien) – zum Angeben des Speicherorts für zusätzliche Dateien, die der Auftrag beim Ausführen des Skripts verwendet.
- Job Parameters (Auftragsparameter) – ein Satz von Schlüssel-Wert-Paaren, die als benannte Parameter an das Auftragsskript übergeben werden. In Python-Aufrufen von AWS Glue API s ist es am besten, Parameter explizit nach Namen zu übergeben. Weitere Hinweise zur Verwendung von Parametern in einem Jobskript finden Sie unter Python-Parameter übergeben und darauf zugreifen in AWS Glue im Entwicklerhandbuch für AWS Glue .
- Tags – Sie können Ihren Auftrag mit Tags organisieren und kenntlich machen.

6. Nachdem Sie die Auftragseigenschaften geändert haben, speichern Sie den Auftrag.

Speichern von Spark-Shuffle-Dateien auf Amazon S3

Einige ETL-Aufträge erfordern das Lesen und Kombinieren von Informationen aus mehreren Partitionen, z. B. bei Verwendung einer Join-Transformation. Dieser Vorgang wird als Mischen (Shuffling) bezeichnet. Beim Mischen werden Daten auf die Festplatte geschrieben und über das Netzwerk übertragen. Mit AWS Glue Version 3.0, Sie können Amazon S3 als Speicherort für diese Dateien konfigurieren. AWS Glue bietet einen Shuffle-Manager, der Shuffle-Dateien in und aus Amazon S3 schreibt und liest. Das Schreiben und Lesen von Shuffle-Dateien aus Amazon S3 ist langsamer (um 5%-20%) als auf einer lokalen Festplatte (oder Amazon EBS, das stark für Amazon optimiert ist). EC2 Amazon S3 bietet jedoch unbegrenzte Speicherkapazität, wodurch Sie nicht mehr mit dem Fehler `No space left on device` (Kein Speicherplatz auf Gerät mehr vorhanden) beim Ausführen Ihres Auftrags rechnen müssen.

Den Auftrag für die Verwendung von Amazon S3 für Misch-Dateien konfigurieren

1. Auf der Seite Jobs (Aufträge) wählen Sie aus der Liste Your jobs (Ihre Aufträge) den Namen des Auftrags aus, den Sie modifizieren möchten.
2. Wählen Sie auf der Seite des visuellen Editors die Registerkarte Job details (Auftragsdetails) aus, die sich oben im Bereich zur Auftragsbearbeitung befindet.

   Scrollen Sie nach unten bis zum Abschnitt Job parameters (Auftragsparameter).
3. Geben Sie die folgenden Schlüssel-Wert-Paare an.

   - `--write-shuffle-files-to-s3` — `true`

Dies ist der Hauptparameter, mit dem der Shuffle-Manager konfiguriert wird AWS Glue um Amazon S3 S3-Buckets zum Schreiben und Lesen von Shuffle-Daten zu verwenden. Dieser Parameter hat standardmäßig den Wert `false`.

- (Optional) `--write-shuffle-spills-to-s3` – `true`

  Mit diesem Parameter können Sie Spill-Dateien in Amazon S3 S3-Buckets auslagern, was Ihrem Spark-Job zusätzliche Stabilität verleiht AWS Glue. Dies ist nur für große Workloads erforderlich, bei denen viele Daten auf die Festplatte übertragen werden. Dieser Parameter hat standardmäßig den Wert `false`.

- (Optional) `--conf spark.shuffle.glue.s3ShuffleBucket` – `S3://<shuffle-bucket>`

  Dieser Parameter gibt den Amazon-S3-Bucket an, der beim Schreiben von Misch-Dateien verwendet werden soll. Wenn Sie diesen Parameter nicht festlegen, ist der Speicherort der Ordner `shuffle-data` am Speicherort für den Temporary path (Temporären Pfad) (`--TempDir`).

> **Note**
>
> Stellen Sie sicher, dass sich der Shuffle-Bucket an derselben Stelle befindet, an der der AWS-Region Job ausgeführt wird.
> Außerdem beseitigt der Misch-Service die Dateien nach der Auftragsausführung nicht. Daher sollten Sie die Richtlinien zum Amazon-S3-Speicherlebenszyklus am Speicherort des Misch-Buckets konfigurieren. Weitere Informationen finden Sie unter Managing your storage lifecycle (Verwaltung des Speicherlebenszyklus) im Amazon-S3-Benutzerhandbuch.

Speichern des Auftrags

Ein roter Hinweis Job has not been saved (Auftrag wurde nicht gespeichert) wird links neben der Schaltfläche Save (Speichern) angezeigt, bis Sie den Auftrag speichern.

Den Auftrag speichern

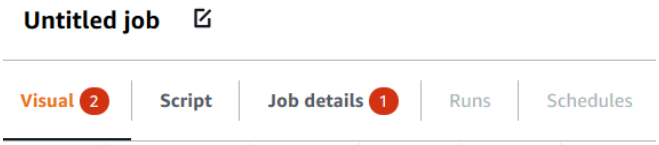
1. Geben Sie alle erforderlichen Informationen in den Registerkarten Visual (Visualisierung) und Job details (Auftragsdetails) an.
2. Wählen Sie die Schaltfläche Save (Speichern) aus.

   Nachdem Sie den Auftrag gespeichert haben, ändert sich der Hinweis mit „not saved (nicht gespeichert)“ und zeigt die Uhrzeit und das Datum an, an dem der Auftrag zuletzt gespeichert wurde.

Wenn Sie den Vorgang beenden AWS Glue Studio bevor Sie Ihren Job speichern, wenn Sie sich das nächste Mal anmelden AWS Glue Studio, wird eine Benachrichtigung angezeigt. Die Benachrichtigung zeigt an, dass ein nicht gespeicherter Auftrag vorhanden ist, und fragt, ob Sie ihn wiederherstellen möchten. Wenn Sie sich entscheiden, den Auftrag wiederherzustellen, können Sie ihn weiter bearbeiten.

Fehlerbehebung beim Speichern eines Auftrags

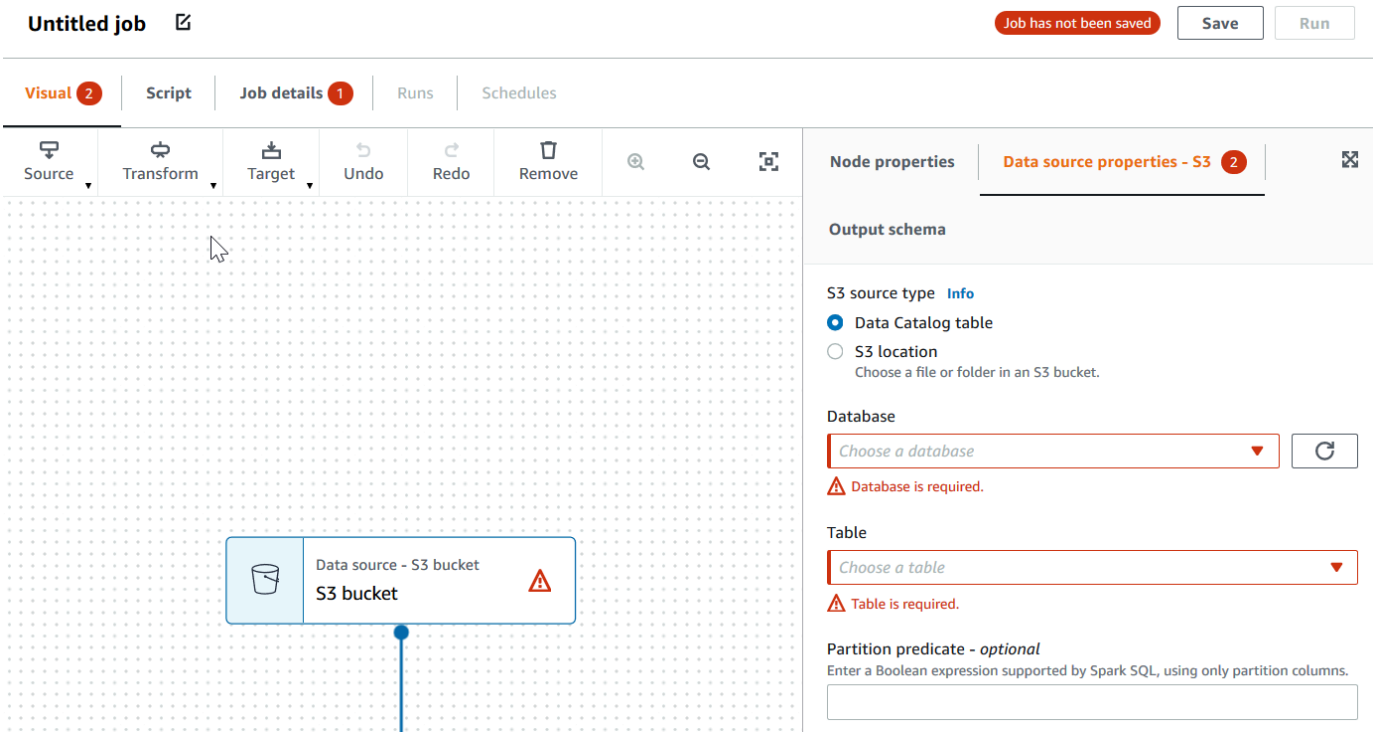
Wenn Sie die Option Save (Speichern) auswählen, aber in Ihrem Auftrag einige erforderliche Informationen fehlen, dann erscheint auf der Registerkarte, auf der die Informationen fehlen, ein roter Hinweis. Die Zahl im Hinweis gibt an, wie viele fehlende Felder erkannt wurden.



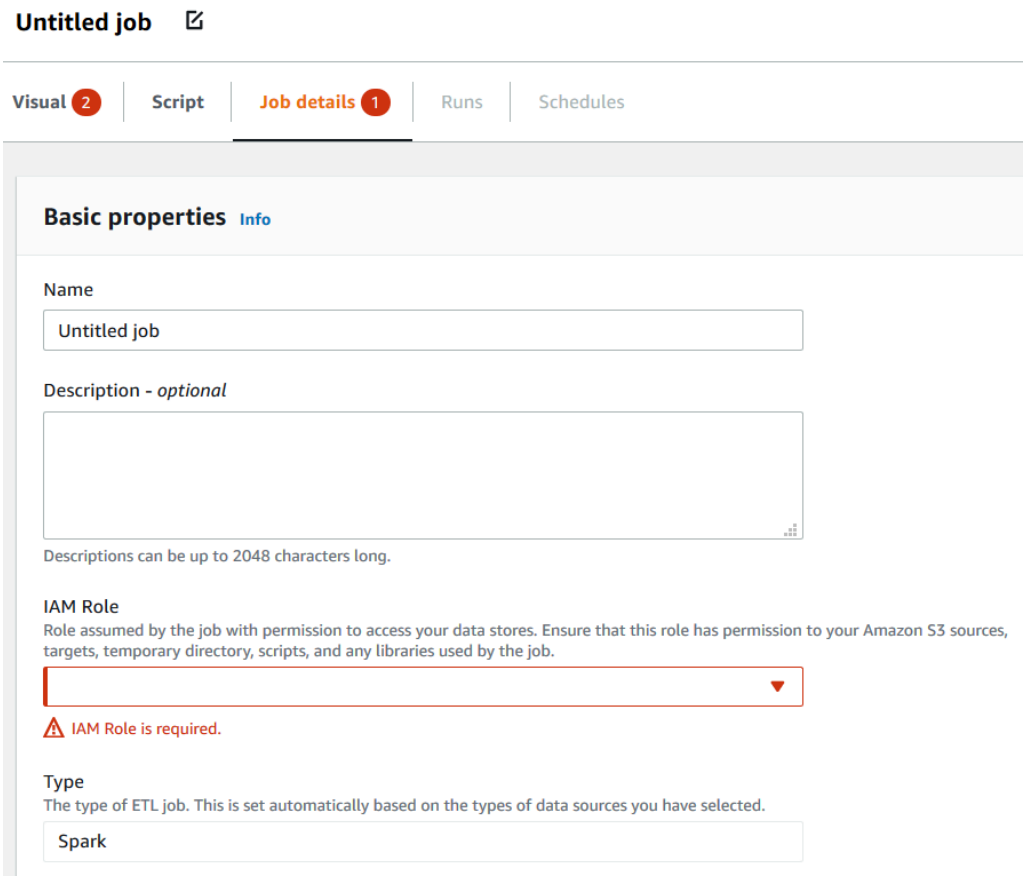
- Wenn ein Knoten im visuellen Editor nicht korrekt konfiguriert ist, erscheint in der Registerkarte Visual (Visualisierung) ein roter Hinweis und der Knoten mit dem Fehler zeigt ein Warnsymbol ⚠

  an.

  1. Wählen Sie den Knoten aus. Im Bereich mit den Knotendetails wird auf der Registerkarte, auf der sich die fehlenden oder falschen Informationen befinden, ein roter Hinweis angezeigt.
  2. Wählen Sie im Bereich mit den Knotendetails die Registerkarte aus, bei der ein roter Hinweis ist, und suchen Sie dann die problematischen Felder, die hervorgehoben sind. Eine Fehlermeldung unter den Feldern enthält zusätzliche Informationen zum Problem.



- Wenn es ein Problem mit den Auftragseigenschaften gibt, erscheint in der Registerkarte Job details (Auftragsdetails) ein roter Hinweis. Wählen Sie diese Registerkarte aus und suchen Sie die problematischen Felder, die hervorgehoben sind. Die Fehlermeldung unter den Feldern enthält zusätzliche Informationen zum Problem.



Klonen von Aufträgen

Sie können mit der Aktion Clone job (Auftrag klonen) einen vorhandenen Auftrag in einen neuen Auftrag kopieren.

Einen neuen Auftrag durch Kopieren eines bestehenden Auftrags erstellen

1. Auf der Seite Jobs (Aufträge) wählen Sie auf der Liste Your jobs (Ihre Aufträge) den Auftrag aus, den Sie duplizieren möchten.
2. Wählen Sie im Menü Actions (Aktionen) die Option Clone job (Auftrag klonen) aus.
3. Geben Sie den Namen des neuen Auftrags ein. Anschließend können Sie den Auftrag speichern oder bearbeiten.

Löschen von Aufträgen

Sie können nicht mehr benötigte Aufträge entfernen. Hierfür können mehrere Aufträge in einem Vorgang gelöscht werden.

Um Jobs zu entfernen von AWS Glue Studio

1. Auf der Seite Jobs (Aufträge) wählen Sie auf der Liste Your jobs (Ihre Aufträge) die Aufträge aus, die Sie löschen möchten.
2. Wählen Sie im Menü Actions (Aktionen) die Option Delete job (Auftrag löschen) aus.
3. Bestätigen Sie, dass Sie den Auftrag löschen möchten, indem Sie **`delete`** eingeben.

Sie können einen gespeicherten Auftrag auch im visuellen Editor auf der Registerkarte Job details (Auftragsdetails) des jeweiligen Auftrags löschen.

Arbeiten mit Aufträgen in AWS Glue

Die folgenden Abschnitte enthalten Informationen über ETL- und Ray-Aufträge in AWS Glue.

Themen

- AWS Glue Versionen
- Arbeiten mit Spark-Jobs in AWS Glue
- Arbeiten mit Ray Jobs in AWS Glue
- Konfiguration von Jobeigenschaften für Python-Shell-Jobs in AWS Glue
- Überwachung AWS Glue

AWS Glue Versionen

Sie können den AWS Glue Versionsparameter konfigurieren, wenn Sie einen Job hinzufügen oder aktualisieren. Die AWS Glue Version bestimmt, welche Versionen von Apache Spark und Python AWS Glue unterstützt werden. Die Python-Version gibt die Version an, die für Aufträge des Typs Spark unterstützt wird. In der folgenden Tabelle sind die verfügbaren AWS Glue -Versionen, die entsprechenden Spark- und Python-Versionen sowie andere Änderungen der Funktionalität aufgeführt.

AWS Glue Versionen

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
AWS Glue 5.0	• Spark 3.5.4 • Python 3.11 • Skala 2.12.18	Java 17	Zusätzlich zu den Framework-Updates sind in diese AWS Glue Version Optimierungen und Upgrades integriert, wie zum Beispiel:

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			• Unterstützung für Amazon SageMaker Unified Studio • Amazon SageMaker Lakehouse-Unterstützung • Open Table Formats (OTF) wurde auf Hudi 0.15.0, Iceberg 1.7.1 und Delta Lake 3.3.0 aktualisiert • Spark-native, feinkörnative Zugriffskontrolle mit Lake Formation. • Unterstützung für Amazon S3 Access Grants • `requirements.txt` Unterstützung für die Installation zusätzlicher Python-Bibliotheken • Unterstützung von Data Lineage in Amazon DataZone

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			• Unterstützung für Amazon S3 Table Bucket • AWS Glue Unterstützung für die Ansicht mehrerer Dialekte im Datenkatalog Einschränkungen In Version 5.0 gibt es folgende Einschränkungen: AWS Glue • Glue Dynamic `GlueContext` Frame/-basierte Zugriffskontrolle auf Tabellene bene mit AWS Lake Formation Berechtigungen, die in Glue 4.0 oder früher unterstützt zt wurden, wird in Glue 5.0 nicht unterstützt. Verwenden Sie die neue native Fine-Grained Access Control (FGAC) von Spark in Glue 5.0.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			Weitere Informationen zur Migration auf Version 5.0 finden Sie AWS Glue unter. Migration AWS Glue für Spark-Jobs auf AWS Glue Version 5.0

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität

AWS Glue 4.0	Versionen der Spark-Umgebung
	• Spark 3.3.0
	• Python 3.10
		Java 8	AWS Glue 4.0 enthält eine Reihe von Optimierungen und Upgrades, die in diese AWS Glue Version integriert sind, wie z. B.:
			• Viele Upgrades der Spark-Funktionalität von Spark 3.1 auf Spark 3.3:
			  • Verschiedene Funktionsverbesserungen bei Kopplung mit Pandas. Weitere Informationen finden Sie unter Neuerungen für Python 3.3.
			  • Zusätzliche Optimierungen, die auf Amazon EMR entwickelt wurden.
			  • Führen Sie ein Upgrade auf EMR File System (EMRFS, EMR-Dateisystem) 2.53 durch.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			• Log4j2-Migration von Log4j1.x • Verschiedene Python-Modul-Aktualisierungen von AWS Glue 3.0, wie z. B. eine aktualisierte Version von Boto. • Upgrade mehrerer Konnektoren, einschließlich des standardmäßigen Amazon-Redshift-Konnektors. Siehe Anhang C: Konnektor-Upgrades. • Upgrade mehrerer JDBC-Treiber. Siehe Anhang B: Aktualisierungen von JDBC-Treibern. • Aktualisiert mit einem neuen Amazon-Redshift-Konnektor und JDBC-Treiber. • Native Unterstützung für Open-Data-Lake-Frameworks mit Apache Hudi,

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			Delta Lake und Apache Iceberg. • Native Unterstützung für das Amazon-S3-basierte Cloud-Shuffle-Speicher-Plugin (ein Apache-Spark-Plugin) zur Verwendung von Amazon S3 für Shuffling und elastische Speicherkapazität. Einschränkungen Dies sind die Einschränkungen für AWS Glue 4.0: • AWS Glue Maschinelles Lernen und Transformationen personenbezogener Daten (PII) sind in Version 4.0 noch nicht verfügbar. AWS Glue Weitere Informationen zur Migration auf

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			AWS Glue Version 4.0 finden Sie unter Migration AWS Glue für Spark-Jobs auf AWS Glue Version 4.0.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
	Versionen für Ray-Umgebungen • Ray 2.4.0 Python 3.9	N/A	Erstellen und führen Sie verteilte Python-Anwendungen mit AWS Glue for Ray aus. • Unterstützt Ray-2.4.0-Datenverteilung (`ray[data]`) mit Python 3.9. Weitere Informationen zu dieser Ray-Version finden Sie unter Ray-2.4.0 im Ray-Repository. GitHub • Unterstützt die Installation zusätzlicher Python-Bibliotheken in der `Ray2.4`-Laufzeitumgebung. Weitere Informationen finden Sie unter the section called “Zusätzliche Python-Module für Ray-Aufträge”. • Integriert Protokolle und Metriken von Ray-Jobs mit Amazon CloudWatc

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			h. Weitere Informationen erhalten Sie unter the section called “Fehlerbehebung bei Ray-Fehlern” und the section called “Auftragsmetriken von Ray”. • Aggregiert und visualisiert Metriken für Ray-Jobs auf AWS Glue Studio jeder Job-Ausführungsseite. • Unterstützt die Verteilung von Dateien an jedes Arbeitsverzeichnis in Ihrem Cluster, das Übertragen von Objekten aus dem Ray-Objektspeicher an Amazon S3 und die Steuerung der Mindestanzahl von Worker-Knoten, die Ihrem Ray-Auftrag zugewiesen sind. Weitere Informationen finden Sie unter the section

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			called “Parameter für Ray-Aufträge”. Einschränkungen für Ray-Aufträge in AWS Glue 4.0 • AWS Glue interaktive Sessions für Ray sind in dieser Version weiterhin als Vorschauversion verfügbar. • AWS Glue für Ray ist die Integration mit Amazon VPC derzeit nicht verfügbar. Ressourcen in einer VPC sind ohne eine öffentliche Route nicht zugänglich. AWS Weitere Informationen zur Verwendung AWS Glue mit Amazon VPC finden Sie unterthe section called “Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			PrivateLink für AWS Glue ”. • AWS Glue for Ray ist in den Ländern USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Tokio) und Europa (Irland) verfügbar.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
AWS Glue 3.0	• Spark 3.1.1 • Python 3.7	Java 8	Zusätzlich zum Upgrade der Spark-Engine auf Version 3.0 gibt es Optimierungen und Upgrades in diesem AWS Glue Release, zum Beispiel: • Erstellt die AWS Glue ETL-Bibliothek auf Grundlage von Spark 3.0, einer Hauptversion für Spark. • Streaming-Aufträge werden in AWS Glue 3.0 unterstützt • Beinhaltet neue AWS Glue Spark-Laufzeitoptimierungen für Leistung und Zuverlässigkeit: • Schnellere spaltenweise Verarbeitung im Speicher basierend auf Apache Arrow zum Lesen von CSV-Daten.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			• SIMD-basierte Ausführung für vektorisierte Lesevorgänge mit CSV-Daten. • Das Spark-Upgrade umfasst weitere Optimierungen, die auf Amazon EMR entwickelt wurden. • Ein Upgrade von EMRFS 2.38 auf 2.46 bietet neue Features und Fehlerbehebungen für den Zugriff auf Amazon S3. • Aktualisierung verschiedener Abhängigkeiten, die für die neue Spark-Version erforderlich waren. • Aktualisierte JDBC-Treiber für unsere nativ unterstützten Datenquellen.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			Einschränkungen Dies sind die Einschränkungen für AWS Glue 3.0: • AWS Glue Transformationen für maschinelles Lernen sind in AWS Glue 3.0 noch nicht verfügbar. • Einige benutzerdefinierte Spark-Konnektors funktionieren nicht für AWS Glue 3.0, wenn sie von Spark 2.4 abhängen und nicht mit Spark 3.1 kompatibel sind.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
AWS Glue 2.0 (Ende der Nutzungsdauer am 1. April 2026)	• Spark 2.4.3 • Python 3.7	N/A	Zusätzlich zu den in AWS Glue Version 1.0 bereitgestellten Funktionen bietet AWS Glue Version 2.0 auch: • Eine verbesserte Infrastruktur für die Ausführung von Apache Spark ETL-Jobs AWS Glue mit kürzeren Startzeit en. • Die Standardp rotokollierung erfolgt jetzt in Echtzeit, mit separaten Streams für Treiber und Ausführende sowie Ausgaben und Fehler. • Support für die Angabe zusätzlic her Python-Module oder verschied ener Versionen auf Auftragsebene.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			Note AWS Glue Version 2.0 unterscheidet sich von AWS Glue Version 1.0 in einigen Abhängigkeiten und Versionen aufgrund der zugrunde liegenden architektonischen Änderungen. Validieren Sie Ihre AWS Glue -Aufträge vor der Migration auf die AWS Glue -Hauptversionen.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
AWS Glue 1.0 (Ende der Nutzungsdauer am 1. April 2026)	• Spark 2.4.3 • Python 2.7 • Python 3.6	N/A	Sie können Auftragslesezeichen für Parquet- und ORC-Formate in AWS Glue -ETL-Aufträgen verwalten (unter Verwendung von AWS Glue Version 1.0). Bisher konnten Sie in AWS Glue ETL-Jobs nur gängige Amazon S3 S3-Quellformate wie JSON, CSV, Apache Avro und XML als Lesezeichen speichern. Bei der Festlegung der Formatoptionen für ETL-Eingaben und -Ausgaben können Sie angeben, dass das Apache reader/writer Avro-Format 1.8 verwendet werden soll, um das Lesen und Schreiben von logischen Avro-Typen zu unterstützen (mit AWS Glue Version 1.0). Bisher wurde nur das reader/wr

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
			iter Avro-Format der Version 1.7 unterstützt. Der DynamoDB-Verbindungstyp unterstützt eine Writer-Option (mit AWS Glue -Version 1.0). Einschränkungen Die folgenden sind die Einschränkungen für AWS Glue 1.0: • AWS Glue Die Versionen 0.9 und 1.0 sind in Zukunft nicht mehr in den Regionen Asien-Pazifik (Jakarta`ap-southeast-3`) (), Naher Osten (VAE`me-central-1`) () oder anderen neuen Regionen verfügbar.

AWS Glue Version	Unterstützte Versionen der Laufzeitumgebung	Unterstützte Java-Version	Änderungen der Funktionalität
AWS Glue 0.9 (Ende der Lebensdauer am 1. April 2026)	• Spark 2.2.1 • Python 2.7	N/A	Jobs, die ohne Angabe einer AWS Glue Version erstellt wurden, sind standardmäßig AWS Glue 0.9. Einschränkungen Die folgenden sind die Einschränkungen für AWS Glue 0.9: • AWS Glue Die Versionen 0.9 und 1.0 sind in Zukunft nicht mehr in den Regionen Asien-Pazifik (Jakarta`ap-southeast-3`) (), Naher Osten (VAE`me-central-1`) () oder anderen neuen Regionen verfügbar .

> **Note**
>
> Die folgenden Glue-Versionen unterstützen diese Versionen von PythonShell:
>
> • PythonShell v3.6 wird in Glue Version 1.0 unterstützt.
>
> • PythonShell v3.9 wird in Glue Version 3.0 unterstützt.

Darüber hinaus werden Entwicklungsendpunkte nur in den Glue-Versionen 1.0 und 0.9 unterstützt.

AWS Glue Richtlinie zur Versionsunterstützung

AWS Glue vist ein serverloser Datenintegrationsdienst, der es einfach macht, Daten für Analytik, Machine Learning und Anwendungsentwicklung zu erkennen, vorzubereiten und zu kombinieren. Ein AWS Glue Job enthält die Geschäftslogik, die die Datenintegrationsarbeit ausführt in AWS Glue. Es gibt drei Arten von Jobs in AWS Glue: Spark (Batch und Streaming), Ray und Python-Shell. Wenn Sie Ihren Job definieren, geben Sie Folgendes an AWS Glue Version, die Versionen in der zugrunde liegenden Spark-, Ray- oder Python-Laufzeitumgebung konfiguriert. Zum Beispiel: ein AWS Glue Version 5.0 Spark Job unterstützt Spark 3.5.4 und Python 3.11.

Support-Richtlinie

AWS Glue Versionen basieren auf einer Kombination aus Betriebssystem, Programmiersprache und Softwarebibliotheken, die Wartungs- und Sicherheitsupdates unterliegen. AWS Glue Die Richtlinie zur Versionsunterstützung sieht vor, dass der Support für eine Version eingestellt wird, wenn für eine wichtige Komponente der Version der Community Long-Term Support (LTS) ausläuft und keine Sicherheitsupdates mehr verfügbar sind. AWS Glue Die Richtlinien zur Versionsunterstützung umfassen die folgenden Status:

Ende des Support (EOS) — Wenn eine AWS Glue Version EOS erreicht:

- AWS Glue wendet keine Sicherheitspatches oder andere Updates mehr auf EOS-Versionen an.
- AWS Glue Jobs mit EOS-Versionen haben keinen Anspruch auf technischen Support.
- AWS Glue wird möglicherweise nicht berücksichtigt SLAs , wenn Jobs auf EOS-Versionen ausgeführt werden.

Ende des Lebenszyklus (EOL) — Wenn eine AWS Glue Version EOL erreicht:

- Sie können keine neuen AWS Glue Jobs oder interaktiven Sitzungen mehr für EOL-Versionen erstellen.
- In diesen AWS Glue Versionen können Sie keine Jobläufe mehr starten.
- AWS Glue beendet bestehende Jobausführungen und interaktive Sitzungen für EOL-Versionen.

- EOL-Versionen werden aus und entfernt AWS Glue SDKs . APIs

Folgendes AWS Glue Versionen haben das Ende des Supports erreicht und sind nach Ablauf der Nutzungsdauer nicht mehr verfügbar. Änderungen am Support-Status einer Version beginnen um Mitternacht (pazifische Zeitzone) am angegebenen Datum.

Typ	Glue Version	Ende des Supports	Ende des Lebens
Spark	Glue Version 0.9 (Spark 2.2, Scala 2, Python 2)	01.06.2022	1.4.2026
Spark	Glue versie 1.0 (Spark 2.4, Python 2)	01.06.2022	1.4.2026
Spark	Glue Version 1.0 (Spark 2.4, Scala 2, Python 3)	30.9.2022	1.4.2026
Spark	Glue Version 2.0 (Spark 2.4, Python 3)	31.1.2024	1.4.2026
Typ	Python-Version	Ende des Supports	Ende des Lebens
Python-Shell	Python 2 (Glue Version 1.0)	01.06.2022	1.4.2026
Python-Shell	PythonShell 3.6 (Glue-Version 1.0)	31.03.2026	N/A
Typ	Notebook-Version	Ende des Supports	Ende des Lebens
Entwicklungsendpunkt	Zeppelin Notebook	30.9.2022	N/A

AWS empfiehlt dringend, dass Sie Ihre Jobs auf unterstützte Versionen migrieren.

Für Informationen zur Migration Ihrer Spark-Jobs auf die neueste Version AWS Glue Version finden Sie unter Migrieren AWS Glue Jobs zu AWS Glue Version 5.0.

Für die Migration Ihrer Python-Shell-Jobs auf die neueste Version AWS Glue Version:

- Wählen Sie in der Konsole `Python 3 (Glue Version 4.0)` aus.
- Setzen Sie in der CreateJob/UpdateJobAPI den `GlueVersion` Parameter auf `2.0` und den Wert `PythonVersion` auf `3` unter dem `Command` Parameter. Die `GlueVersion` Konfiguration hat keinen Einfluss auf das Verhalten von Python-Shell-Jobs, sodass das Inkrementieren `GlueVersion` keinen Vorteil hat.
- Sie müssen Ihr Auftragsskript mit Python 3 kompatibel machen.

Migration AWS Glue für Spark-Jobs auf AWS Glue Version 5.0

In diesem Thema werden die Änderungen zwischen den AWS Glue Versionen 0.9, 1.0, 2.0, 3.0 und 4.0 beschrieben, die es Ihnen ermöglichen, Ihre Spark-Anwendungen und ETL-Jobs auf AWS Glue 5.0 zu migrieren. Außerdem werden die Funktionen in AWS Glue 5.0 und die Vorteile seiner Verwendung beschrieben.

Um diese Funktion mit Ihren AWS Glue ETL-Jobs zu verwenden, wählen Sie **`5.0`** `Glue version` bei der Erstellung Ihrer Jobs die Option.

Themen

- Neue Features
- Aktionen AWS Glue zur Migration auf 5.0
- Checkliste für die Migration
- AWS Glue 5.0-Funktionen
- Migration von AWS Glue 4.0 auf 5.0 AWS Glue
- Migration von AWS Glue 3.0 auf 5.0 AWS Glue
- Migration von AWS Glue 2.0 auf 5.0 AWS Glue
- Verhaltensänderungen in 5.0 protokollieren AWS Glue
- Migration von Konnektoren und JDBC-Treibern für 5.0 AWS Glue

Neue Features

In diesem Abschnitt werden die neuen Funktionen und Vorteile von AWS Glue Version 5.0 beschrieben.

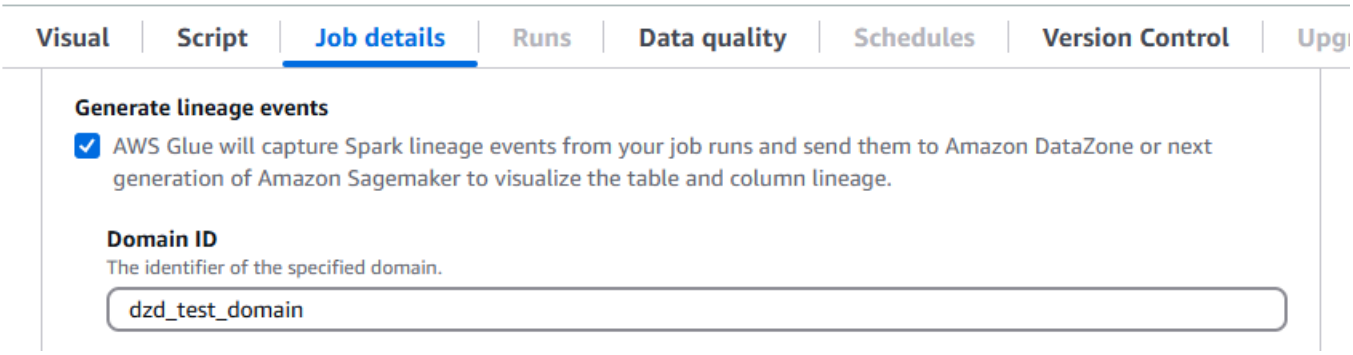
- Apache Spark-Update von 3.3.0 in AWS Glue 4.0 auf 3.5.4 in AWS Glue 5.0. Siehe Wesentliche Verbesserungen von Spark 3.3.0 auf Spark 3.5.4.
- Spark-native Fine-Grained Access Control (FGAC) mit Lake Formation. Dazu gehört FGAC für Iceberg-, Delta- und Hudi-Tabellen. Weitere Informationen finden Sie unter Verwenden von AWS Glue with AWS Lake Formation für eine detaillierte Zugriffskontrolle.

  Beachten Sie die folgenden Überlegungen oder Einschränkungen für Spark-native FGAC:

  - Derzeit werden Datenschreibvorgänge nicht unterstützt
  - Um `GlueContext` mithilfe von Lake Formation in Iceberg zu schreiben, muss stattdessen die IAM-Zugriffskontrolle verwendet werden

  Eine vollständige Liste der Einschränkungen und Überlegungen bei der Verwendung von Spark-nativem FGAC finden Sie unter. the section called “Überlegungen”
- Support für Amazon S3 Access Grants als skalierbare Zugriffskontrolllösung für Ihre Amazon S3 S3-Daten von AWS Glue. Weitere Informationen finden Sie unter Verwenden von Amazon S3 Access Grants mit AWS Glue.
- Open Table Formats (OTF) wurde auf Hudi 0.15.0, Iceberg 1.7.1 und Delta Lake 3.3.0 aktualisiert
- Unterstützung für Amazon SageMaker Unified Studio.
- Amazon SageMaker Lakehouse und Integration der Datenabstraktion. Weitere Informationen finden Sie unter Abfragen von Metastore-Datenkatalogen über ETL AWS Glue.
- Support für die Installation zusätzlicher Python-Bibliotheken mit`requirements.txt`. Weitere Informationen finden Sie unter Installation zusätzlicher Python-Bibliotheken in AWS Glue 5.0 mit requirements.txt.
- AWS Glue 5.0 unterstützt Data Lineage in Amazon DataZone. Sie können so konfigurieren AWS Glue , dass während der Ausführung von Spark-Jobs automatisch Herkunftsinformationen gesammelt und die Herkunftsereignisse zur Visualisierung in Amazon gesendet werden. DataZone Weitere Informationen finden Sie unter Data Lineage in Amazon DataZone.

  Um dies in der AWS Glue Konsole zu konfigurieren, aktivieren Sie Generate Lineage Events und geben Sie Ihre DataZone Amazon-Domain-ID auf der Registerkarte Jobdetails ein.



Alternativ können Sie den folgenden Job-Parameter angeben (geben Sie Ihre DataZone Domain-ID an):

- Schlüssel: `--conf`
- Wert:

```
extraListeners=io.openlineage.spark.agent.OpenLineageSparkListener
—conf spark.openlineage.transport.type=amazon_datazone_api
-conf spark.openlineage.transport.domainId=<your-domain-ID>
```

- Konnektor- und JDBC-Treiber-Updates. Weitere Informationen erhalten Sie unter Anhang B: Aktualisierungen von JDBC-Treibern und Anhang C: Konnektor-Upgrades.
- Java-Update von 8 auf 17.
- Erhöhung des Speicherplatzes für `G.2X` Mitarbeiter AWS Glue `G.1X` und Erhöhung des Festplattenspeichers auf 94 GB bzw. 138 GB. Darüber hinaus sind neue Worker-Typen und speicheroptimierte `G.12X``G.16X`,`R.1X`, `R.2X``R.4X`, in AWS Glue Version 4.0 und `R.8X` späteren Versionen verfügbar. Weitere Informationen finden Sie unter Aufträge.
- Jobs, die AWS SDK for Java, Version AWS Glue 2-5.0 Support, können die Java-Versionen 1.12.569 oder 2.28.8 verwenden, wenn der Job v2 unterstützt. Das AWS SDK for Java 2.x ist eine umfassende Neufassung der Codebasis von Version 1.x. Sie basiert auf Java 8+ und fügt mehrere häufig angeforderte Funktionen hinzu. Dies umfasst die Unterstützung von blockierungsfreiem E/A und die Möglichkeit, zur Laufzeit eine andere HTTP-Implementierung einzubinden. Weitere Informationen, einschließlich eines Migrationshandbuchs von SDK for Java v1 zu v2, finden Sie im Handbuch AWS SDK for Java, Version 2.
-

Abwärtskompatible Änderungen

Beachten Sie die folgenden grundlegenden Änderungen:

- Wenn in AWS Glue 5.0 das S3A-Dateisystem verwendet wird und sowohl `fs.s3a.endpoint` als auch `fs.s3a.endpoint.region` nicht gesetzt sind, ist die von S3A verwendete Standardregion `us-east-2`. Dies kann zu Problemen führen, z. B. zu Timeout-Fehlern beim S3-Upload, insbesondere bei VPC-Jobs. Um die durch diese Änderung verursachten Probleme zu beheben, legen Sie die Spark-Konfiguration `fs.s3a.endpoint.region` fest, wenn Sie das S3A-Dateisystem in 5.0 verwenden. AWS Glue
- Feinkörnige Zugangskontrolle (FGAC) von Lake Formation
  - AWS Glue 5.0 unterstützt nur das neue Spark-native FGAC, das Spark verwendet. DataFrames Die Verwendung von FGAC wird nicht unterstützt. AWS Glue DynamicFrames
    - Die Verwendung von FGAC in 5.0 erfordert eine Migration von zu Spark AWS Glue DynamicFrames DataFrames
    - Wenn Sie FGAC nicht benötigen, ist es nicht notwendig, zu Spark zu migrieren, DataFrame und GlueContext Funktionen wie Job-Lesezeichen und Push-Down-Prädikate funktionieren weiterhin.
  - Jobs mit Spark-nativem FGAC erfordern mindestens 4 Mitarbeiter: einen Benutzertreiber, einen Systemtreiber, einen System-Executor und einen Standby-User-Executor.
  - Weitere Informationen finden Sie unter Using with für eine detaillierte Zugriffskontrolle. AWS GlueAWS Lake Formation
- Lake Formation Vollständiger Tabellenzugriff (FTA)
  - AWS Glue 5.0 unterstützt FTA mit Spark-Native DataFrames (neu) und GlueContext DynamicFrames (Legacy, mit Einschränkungen)
  - Spark-natives FTA
    - Wenn das 4.0-Skript verwendet wird GlueContext, wechseln Sie zur Verwendung von nativem Spark.
    - Diese Funktion ist auf Hive- und Iceberg-Tabellen beschränkt
    - Weitere Informationen zur Konfiguration eines 5.0-Jobs für die Verwendung von Spark Native FTA finden Sie unter
  - GlueContext DynamicFrame FTA
    - Keine Codeänderung erforderlich
    - Diese Funktion ist auf Nicht-OTF-Tabellen beschränkt — sie funktioniert nicht mit Iceberg, Delta Lake und Hudi.
- Der vektorisierte SIMD CSV-Reader wird nicht unterstützt.

- Die kontinuierliche Protokollierung in der Ausgabeprotokollgruppe wird nicht unterstützt. Verwenden Sie stattdessen die `error` Protokollgruppe.
- Der AWS Glue Job Run Insights `job-insights-rule-driver` ist veraltet. Der `job-insights-rca-driver` Protokollstream befindet sich jetzt in der Fehlerprotokollgruppe.
- Athena-basierte custom/marketplace Konnektoren werden nicht unterstützt.
- Adobe Marketo Engage, Facebook Ads, Google Ads, Google Analytics 4, Google Sheets, Hubspot, Instagram Ads, Intercom, Jira Cloud, Oracle, Salesforce, Salesforce Marketing Cloud NetSuite, Salesforce Marketing Cloud Account Engagement, SAP,, Slack, Snapchat Ads, Stripe OData ServiceNow, Zendesk und Zoho CRM Connectors werden nicht unterstützt.
- Benutzerdefinierte log4j-Eigenschaften werden in 5.0 nicht unterstützt. AWS Glue

Wesentliche Verbesserungen von Spark 3.3.0 auf Spark 3.5.4

Berücksichtigen Sie die folgenden Verbesserungen:

- Python-Client für Spark Connect (SPARK-39375).
- Implementieren Sie die Unterstützung für DEFAULT-Werte für Spalten in Tabellen (SPARK-38334).
- Support „Lateral Column Alias References“ (SPARK-27561).
- Versichern Sie die Verwendung von SQLSTATE für Fehlerklassen (SPARK-41994).
- Aktivieren Sie standardmäßig den Bloom-Filter Joins (SPARK-38841).
- Bessere Skalierbarkeit und Treiberstabilität der Spark-Benutzeroberfläche für große Anwendungen (SPARK-41053).
- Asynchrone Fortschrittsverfolgung beim strukturierten Streaming (SPARK-39591).
- Beliebige zustandsbehaftete Python-Verarbeitung in strukturiertem Streaming (SPARK-40434).
- Verbesserungen der Pandas-API-Abdeckung (SPARK-42882) und Eingabeunterstützung in (SPARK-39405). NumPy PySpark
- Stellen Sie einen Speicherprofiler für benutzerdefinierte Funktionen bereit (SPARK-40281). PySpark
- Implementieren Sie PyTorch den Verteiler (SPARK-41589).
- Veröffentlichen Sie SBOM-Artefakte (SPARK-41893).
- IPv6Nur-Unterstützungsumgebung (SPARK-39457).
- Kundenspezifischer K8s-Scheduler (Apache YuniKorn und Volcano) GA (SPARK-42802).
- Unterstützung für Scala- und Go-Clients in Spark Connect (SPARK-42554) und (SPARK-43351).

- PyTorchbasierte verteilte ML-Unterstützung für Spark Connect (SPARK-42471).
- Strukturierte Streaming-Unterstützung für Spark Connect in Python und Scala (SPARK-42938).
- Pandas-API-Unterstützung für den Python Spark Connect Client (SPARK-42497).
- Stellen Sie Arrow Python UDFs (SPARK-40307) vor.
- Support benutzerdefinierte Tabellenfunktionen in Python (SPARK-43798).
- Migrieren Sie PySpark Fehler auf Fehlerklassen (SPARK-42986).
- PySpark Testframework (SPARK-44042).
- Unterstützung für HllSketch Datasketches hinzugefügt (SPARK-16484).
- Integrierte Verbesserung der SQL-Funktionen (SPARK-41231).
- IDENTIFIER-Klausel (SPARK-43205).
- Fügen Sie SQL-Funktionen zur Scala-, Python- und R-API hinzu (SPARK-43907).
- Fügen Sie Unterstützung für benannte Argumente für SQL-Funktionen hinzu (SPARK-43922).
- Vermeiden Sie unnötige Wiederholungen von Aufgaben auf dem außer Betrieb genommenen Executor, der verloren geht, wenn Shuffle-Daten migriert werden (SPARK-41469).
- Verteiltes ML <> Spark Connect (SPARK-42471).
- DeepSpeed Verteiler (SPARK-44264).
- Implementieren Sie das Changelog-Checkpointing für den RocksDB-Statusspeicher (SPARK-43421).
- Führen Sie die Ausbreitung von Wasserzeichen zwischen Operatoren ein (SPARK-42376).
- Führen Sie Watermark dropDuplicatesWithin (SPARK-42931) ein.
- Verbesserungen der Speicherverwaltung für den RocksDB State Store Provider (SPARK-43311).

Aktionen AWS Glue zur Migration auf 5.0

Ändern Sie bei vorhandenen Aufträgen die `Glue version` von der vorherigen Version auf `Glue 5.0` in der Auftragskonfiguration.

- Wählen Sie in AWS Glue Studio `Glue 5.0 - Supports Spark 3.5.4, Scala 2, Python 3` in`Glue version`.
- Wählen Sie in der API **`5.0`** im `GlueVersion`-Parameter in der `UpdateJob`-API-Operation aus.

Wählen Sie für neue Aufträge `Glue 5.0` aus, wenn Sie Aufträge erstellen.

- Wählen Sie in der Konsole `Spark 3.5.4, Python 3 (Glue Version 5.0) or Spark 3.5.4, Scala 2 (Glue Version 5.0)` in der `Glue version` aus.
- Wählen Sie in AWS Glue Studio `Glue 5.0 - Supports Spark 3.5.4, Scala 2, Python 3` in`Glue version`.
- Wählen Sie in der API **`5.0`** im `GlueVersion`-Parameter in der `CreateJob`-API-Operation aus.

Um Spark-Ereignisprotokolle von AWS Glue 5.0 aus Version AWS Glue 2.0 oder früher anzuzeigen, starten Sie einen aktualisierten Spark-Verlaufsserver für AWS Glue 5.0 mit AWS CloudFormation oder Docker.

Checkliste für die Migration

Überprüfen Sie diese Checkliste für die Migration:

- Java 17-Aktualisierungen
- [Scala] Aktualisieren Sie AWS SDK-Aufrufe von Version 1 auf Version 2
- Migration von Python 3.10 nach 3.11
- [Python] Aktualisieren Sie die Boto-Referenzen von 1.26 auf 1.34

AWS Glue 5.0-Funktionen

In diesem Abschnitt werden die AWS Glue Funktionen ausführlicher beschrieben.

Abfragen von Metastore-Datenkatalogen über ETL AWS Glue

Sie können Ihren AWS Glue Job registrieren, um auf die zuzugreifen AWS Glue Data Catalog, wodurch Tabellen und andere Metastore-Ressourcen unterschiedlichen Benutzern zur Verfügung stehen. Der Datenkatalog unterstützt eine Hierarchie mit mehreren Katalogen, die all Ihre Daten über Amazon S3 S3-Datenseen hinweg vereinheitlicht. Es bietet außerdem sowohl eine Hive-Metastore-API als auch eine Open-Source-Apache Iceberg-API für den Zugriff auf die Daten. Diese Funktionen sind für AWS Glue und andere datenorientierte Dienste wie Amazon EMR, Amazon Athena und Amazon Redshift verfügbar.

Wenn Sie Ressourcen im Datenkatalog erstellen, können Sie von jeder SQL-Engine aus darauf zugreifen, die die Apache Iceberg REST-API unterstützt. AWS Lake Formation verwaltet Berechtigungen. Nach der Konfiguration können Sie die Funktionen zur Abfrage unterschiedlicher Daten nutzen AWS Glue, indem Sie diese Metastore-Ressourcen mit vertrauten Anwendungen abfragen. Dazu gehören Apache Spark und Trino.

Wie sind Metadaten-Ressourcen organisiert

Daten werden in einer logischen Hierarchie von Katalogen, Datenbanken und Tabellen organisiert, wobei Folgendes verwendet wird AWS Glue Data Catalog:

- Katalog — Ein logischer Container, der Objekte aus einem Datenspeicher enthält, z. B. Schemas oder Tabellen.
- Datenbank — Organisiert Datenobjekte wie Tabellen und Ansichten in einem Katalog.
- Tabellen und Ansichten — Datenobjekte in einer Datenbank, die eine Abstraktionsebene mit einem verständlichen Schema bieten. Sie erleichtern den Zugriff auf zugrunde liegende Daten, die in verschiedenen Formaten und an verschiedenen Orten vorliegen können.

Migration von AWS Glue 4.0 auf 5.0 AWS Glue

Alle bestehenden Jobparameter und wichtigen Funktionen, die in AWS Glue 4.0 vorhanden sind, werden in AWS Glue 5.0 verfügbar sein, mit Ausnahme von Transformationen für maschinelles Lernen.

Die folgenden neuen Parameter wurden hinzugefügt:

- `--enable-lakeformation-fine-grained-access`: Aktiviert die FGAC-Funktion (Fine-Grained Access Control) in AWS Lake Formation-Tabellen.

Weitere Informationen finden Sie in der Dokumentation zur Spark-Migration:

- Migrationshandbuch: Spark Core
- Migrationshandbuch: SQL, Datensätze und DataFrame
- Migrationshandbuch: Strukturiertes Streaming
- Wird upgegradet PySpark

Migration von AWS Glue 3.0 auf 5.0 AWS Glue

> **Note**
>
> Informationen zu Migrationsschritten im Zusammenhang mit AWS Glue 4.0 finden Sie unterMigration von AWS Glue 3.0 auf 4.0 AWS Glue.

Alle bestehenden Jobparameter und Hauptfunktionen, die in AWS Glue 3.0 vorhanden sind, werden auch in AWS Glue 5.0 verfügbar sein, mit Ausnahme von Transformationen für maschinelles Lernen.

Migration von AWS Glue 2.0 auf 5.0 AWS Glue

> **Note**
>
> Migrationsschritte im Zusammenhang mit AWS Glue 4.0 und eine Liste der Migrationsunterschiede zwischen AWS Glue Version 3.0 und 4.0 finden Sie unterMigration von AWS Glue 3.0 auf 4.0 AWS Glue.

Beachten Sie auch die folgenden Migrationsunterschiede zwischen den AWS Glue Versionen 3.0 und 2.0:

- Alle bestehenden Jobparameter und Hauptfunktionen, die es in AWS Glue 2.0 gibt, werden auch in AWS Glue 5.0 verfügbar sein, mit Ausnahme von Transformationen für maschinelles Lernen.
- Einige Spark-Änderungen allein erfordern möglicherweise eine Überarbeitung Ihrer Skripte, um sicherzustellen, dass entfernte Features nicht referenziert werden. Zum Beispiel aktiviert Spark 3.1.1 und höher Scala-untyped nicht, UDFs aber Spark 2.4 erlaubt sie.
- Python 2.7 wird nicht unterstützt.
- Alle zusätzlichen JAR-Dateien, die in bestehenden AWS Glue 2.0-Jobs bereitgestellt werden, können zu widersprüchlichen Abhängigkeiten führen, da es Upgrades in mehreren Abhängigkeiten gab. Sie können Klassenpfadkonflikte mit dem Job-Parameter vermeiden. `--user-jars-first`
- Änderungen am Verhalten von loading/saving Parquet-Dateien mit Zeitstempel from/to . Weitere Informationen finden Sie unter Upgrade von Spark SQL 3.0 auf 3.1.
- Unterschiedliche Parallelität der Spark-Tasks für die Konfiguration. driver/executor Sie können die Aufgabenparallelität anpassen, indem Sie das Job-Argument übergeben. `--executor-cores`

Verhaltensänderungen in 5.0 protokollieren AWS Glue

Im Folgenden sind die Änderungen des Protokollierungsverhaltens in AWS Glue 5.0 aufgeführt. Weitere Informationen finden Sie unter Protokollierung von AWS Glue Aufträgen.

- Alle Protokolle (Systemprotokolle, Spark-Daemon-Protokolle, Benutzerprotokolle und Glue Logger-Protokolle) werden jetzt standardmäßig in die `/aws-glue/jobs/error` Protokollgruppe geschrieben.

- Die `/aws-glue/jobs/logs-v2` Protokollgruppe, die in früheren Versionen für die kontinuierliche Protokollierung verwendet wurde, wird nicht mehr verwendet.
- Sie können die Namen der Protokollgruppe oder des Protokolldatenstroms nicht mehr mithilfe der entfernten Argumente für die kontinuierliche Protokollierung umbenennen oder anpassen. Sehen Sie sich stattdessen die neuen Job-Argumente in AWS Glue 5.0 an.

In AWS Glue 5.0 wurden zwei neue Job-Argumente eingeführt

- `--custom-logGroup-prefix`: Ermöglicht es Ihnen, ein benutzerdefiniertes Präfix für die Gruppen `/aws-glue/jobs/error` und `/aws-glue/jobs/output` Logs anzugeben.
- `--custom-logStream-prefix`: Ermöglicht es Ihnen, ein benutzerdefiniertes Präfix für die Log-Stream-Namen innerhalb der Log-Gruppen anzugeben.

  Zu den Validierungsregeln und Einschränkungen für benutzerdefinierte Präfixe gehören:

  - Der gesamte Protokollstreamname muss zwischen 1 und 512 Zeichen lang sein.
  - Das benutzerdefinierte Präfix für Log-Stream-Namen ist auf 400 Zeichen begrenzt.
  - Zu den zulässigen Zeichen in Präfixen gehören alphanumerische Zeichen, Unterstriche (`_`), Bindestriche (`-`) und Schrägstriche (`/`).

Argumente für AWS Glue die kontinuierliche Protokollierung wurden in 5.0 nicht mehr unterstützt

Die folgenden Job-Argumente für die kontinuierliche Protokollierung sind in 5.0 veraltet AWS Glue

- `--enable-continuous-cloudwatch-log`
- `--continuous-log-logGroup`
- `--continuous-log-logStreamPrefix`
- `--continuous-log-conversionPattern`
- `--enable-continuous-log-filter`

Migration von Konnektoren und JDBC-Treibern für 5.0 AWS Glue

Die aktualisierten Versionen von JDBC- und Data-Lake-Konnektoren finden Sie unter:

- Anhang B: Aktualisierungen von JDBC-Treibern
- Anhang C: Konnektor-Upgrades

- Anhang D: Upgrades im Open-Table-Format

Die folgenden Änderungen gelten für die Konnektor- oder Treiberversionen, die in den Anhängen für Glue 5.0 aufgeführt sind.

Amazon Redshift

Beachten Sie folgende Änderungen:

- Integriert die Unterstützung für dreiteilige Tabellennamen, damit der Connector Redshift-Datenaustauschtabellen abfragen kann.
- Korrigiert die Zuordnung von Spark `ShortType` dahingehend, dass Redshift verwendet wird, `SMALLINT` anstatt der erwarteten Datengröße besser gerecht `INTEGER` zu werden.
- Unterstützung für benutzerdefinierte Clusternamen (CNAME) für Amazon Redshift Serverless hinzugefügt.

Apache Hudi

Beachten Sie folgende Änderungen:

- Support den Index auf Rekordniveau.
- Support auto Generierung von Datensatzschlüsseln. Jetzt müssen Sie das Datensatzschlüsselfeld nicht mehr angeben.

Apache Iceberg

Beachten Sie folgende Änderungen:

- Support Sie eine feinkörnige Zugriffskontrolle mit. AWS Lake Formation
- Support Branching und Tagging, bei denen es sich um benannte Verweise auf Snapshots mit eigenen unabhängigen Lebenszyklen handelt.
- Es wurde eine Changelog-Ansichtsprozedur hinzugefügt, die eine Ansicht generiert, die die Änderungen enthält, die in einem bestimmten Zeitraum oder zwischen bestimmten Snapshots an einer Tabelle vorgenommen wurden.

Delta Lake

Beachten Sie folgende Änderungen:

- Support Delta Universal Format (UniForm), das einen nahtlosen Zugriff über Apache Iceberg und Apache Hudi ermöglicht.
- Support Löschvektoren, die ein Merge-on-Read Paradigma implementieren.

AzureCosmos

Beachten Sie folgende Änderungen:

- Unterstützung für hierarchische Partitionsschlüssel hinzugefügt.
- Option hinzugefügt, um ein benutzerdefiniertes Schema mit StringType (rohes JSON) für eine verschachtelte Eigenschaft zu verwenden.
- Es wurde eine Konfigurationsoption hinzugefügt`spark.cosmos.auth.aad.clientCertPemBase64`, um die Verwendung der SPN-Authentifizierung (ServicePrincipal Name) mit Zertifikat anstelle des geheimen Client-Schlüssels zu ermöglichen.

Weitere Informationen finden Sie im Änderungsprotokoll des Azure Cosmos DB Spark-Connectors.

Microsoft SQL Server

Beachten Sie folgende Änderungen:

- Die TLS-Verschlüsselung ist standardmäßig aktiviert.
- Wenn encrypt = false ist, der Server aber eine Verschlüsselung benötigt, wird das Zertifikat anhand der `trustServerCertificate` Verbindungseinstellung validiert.
- `aadSecurePrincipalId`und `aadSecurePrincipalSecret` veraltet.
- `getAADSecretPrincipalId`API wurde entfernt.
- CNAME-Auflösung hinzugefügt, wenn Realm angegeben ist.

MongoDB

Beachten Sie folgende Änderungen:

- Support für den Micro-Batch-Modus mit Spark Structured Streaming.

- Support für BSON-Datentypen.
- Unterstützung für das Lesen mehrerer Sammlungen im Micro-Batch- oder Continuous-Streaming-Modus wurde hinzugefügt.
  - Wenn der Name einer Sammlung, die in Ihrer `collection` Konfigurationsoption verwendet wird, ein Komma enthält, behandelt der Spark-Connector sie als zwei verschiedene Sammlungen. Um dies zu vermeiden, müssen Sie das Komma umgehen, indem Sie ihm einen umgekehrten Schrägstrich (\) voranstellen.
  - Wenn der Name einer Sammlung, die in Ihrer `collection` Konfigurationsoption verwendet wird, „*“ ist, interpretiert der Spark-Connector dies als Spezifikation zum Scannen aller Sammlungen. Um dies zu vermeiden, müssen Sie das Sternchen maskieren, indem Sie ihm einen umgekehrten Schrägstrich (\) voranstellen.
  - Wenn der Name einer Sammlung, die in Ihrer `collection` Konfigurationsoption verwendet wird, einen umgekehrten Schrägstrich (\) enthält, behandelt der Spark-Connector den umgekehrten Schrägstrich als Escape-Zeichen, was die Interpretation des Werts ändern kann. Um dies zu vermeiden, müssen Sie den Backslash umgehen, indem Sie ihm einen weiteren Backslash voranstellen.

Weitere Informationen finden Sie in den Versionshinweisen zum MongoDB-Konnektor für Spark.

Snowflake

Beachten Sie folgende Änderungen:

- Es wurde ein neuer `trim_space` Parameter eingeführt, mit dem Sie Werte von `StringType` Spalten beim Speichern in einer Snowflake-Tabelle automatisch kürzen können. Standard: `false`.
- Der `abort_detached_query` Parameter wurde standardmäßig auf Sitzungsebene deaktiviert.
- Die Anforderung des `SFUSER` Parameters bei der Verwendung von OAUTH wurde entfernt.
- Die Funktion Advanced Query Pushdown wurde entfernt. Alternativen zu dieser Funktion sind verfügbar. Anstatt beispielsweise Daten aus Snowflake-Tabellen zu laden, können Benutzer Daten direkt aus Snowflake-SQL-Abfragen laden.

Weitere Informationen finden Sie in den Versionshinweisen zu Snowflake Connector for Spark.

Anhang A: Nennenswerte Aktualisierungen von Abhängigkeiten

Im Folgenden sind Abhängigkeits-Upgrades aufgeführt:

-Abhängigkeit	Version in 5.0 AWS Glue	Version in AWS Glue 4.0	Version in AWS Glue 3.0	Version in AWS Glue 2.0	Version in AWS Glue 1.0
Java	17	8	8	8	8
Spark	3.5.4	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3
Hadoop	3.4.1	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Scala	2.12,18	2.12	2.12	2.11	2.11
Jackson	2.15,2	2.12	2.12	2.11	2.11
Hive	2.3.9-amzn-4	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2,69,0	2,54,0	2,46,0	2.38.0	2.30.0
JSON4s	3.7.0-M11	3.7.0-M11	3.6.6	3.5.x	3.5.x
Arrow	12.0.1	7.0.0	2.0.0	0.10.0	0.10.0
AWS Glue Datenkatalog-Client	4.5.0	3.7.0	3.0.0	1.10.0	N/A
AWS SDK for Java	2.29.52	1.12	1.12		
Python	3,11	3,10	3.7	2.7 und 3.6	2.7 und 3.6
Boto	1,34,131	1,26	1,18	1.12	N/A
EMR DynamoDB-Anschluss	5.6.0	4.16.0			

Anhang B: Aktualisierungen von JDBC-Treibern

Die folgenden JDBC-Treiber-Upgrades sind:

Treiber	JDBC-Treiberversion in 5.0 AWS Glue	JDBC-Treiberversion in 4.0 AWS Glue	JDBC-Treiberversion in 3.0 AWS Glue	JDBC-Treiberversion in früheren Versionen AWS Glue
MySQL	8.0.33	8.0.23	8.0.23	5.1
Microsoft SQL Server	10.2,0	9.4,0	7.0.0	6.1.0
Oracle-Datenbanken	23.3.0.23,09	21,7	21,1	11.2
PostgreSQL	42,7,3	42,3,6	42,2,18	42,10
Amazon Redshift	Rotshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017
SAP Hana	2.20.17	2.17,12		
Teradata	20,00,00,33	20.00.00.06		

Anhang C: Konnektor-Upgrades

Im Folgenden sind Konnektor-Upgrades aufgeführt:

Treiber	Connector-Version in 5.0 AWS Glue	Connector-Version in AWS Glue 4.0	Connector-Version in AWS Glue 3.0
EMR DynamoDB-Anschluss	5.6.0	4.16.0	
Amazon Redshift	6.4.0	6.1.3	
OpenSearch	1.2.0	1.0.1	
MongoDB	10.4.0	10.0.4	3.0.0

Treiber	Connector-Version in 5.0 AWS Glue	Connector-Version in AWS Glue 4.0	Connector-Version in AWS Glue 3.0
Snowflake	3.0.0	2.12.0	
Google BigQuery	0,32,2	0,32,2	
AzureCosmos	4,33,0	4,22,0	
AzureSQL	1.3.0	1.3.0	
Vertica	3.3.5	3.3.5	

Anhang D: Upgrades im Open-Table-Format

Im Folgenden finden Sie Upgrades im Open-Table-Format:

OTF	Connector-Version in AWS Glue 5.0	Connector-Version in AWS Glue 4.0	Connector-Version in AWS Glue 3.0
Hudi	0.15.0	0.12.1	0.10.1
Delta Lake	3.3.0	2.1.0	1.0.0
Iceberg	1.7.1	1.0.0	0.13.1

Migration AWS Glue für Spark-Jobs auf AWS Glue Version 4.0

In diesem Thema werden die Änderungen zwischen den AWS Glue Versionen 0.9, 1.0, 2.0 und 3.0 beschrieben, die es Ihnen ermöglichen, Ihre Spark-Anwendungen und ETL-Jobs auf AWS Glue 4.0 zu migrieren. Außerdem werden die Funktionen von AWS Glue 4.0 und die Vorteile ihrer Verwendung beschrieben.

Um diese Funktion mit Ihren AWS Glue ETL-Jobs zu verwenden, wählen Sie **`4.0`** `Glue version` bei der Erstellung Ihrer Jobs die Option.

Themen

- Neue unterstützte Features

- Aktionen zur Migration zu AWS Glue 4.0
- Checkliste für die Migration
- Migration von AWS Glue 3.0 auf 4.0 AWS Glue
- Migration von 2.0 auf 4.0 AWS GlueAWS Glue
- Migration von AWS Glue 1.0 auf 4.0 AWS Glue
- Migration von AWS Glue 0.9 auf 4.0 AWS Glue
- Migration von Connector und JDBC-Treibern für 4.0 AWS Glue
- Anhang A: Nennenswerte Aktualisierungen von Abhängigkeiten
- Anhang B: Aktualisierungen von JDBC-Treibern
- Anhang C: Konnektor-Upgrades

Neue unterstützte Features

In diesem Abschnitt werden die neuen Funktionen und Vorteile von AWS Glue Version 4.0 beschrieben.

- Es basiert auf Apache Spark 3.3.0, enthält jedoch Optimierungen in AWS Glue und Amazon EMR, wie z. B. adaptive Abfrageläufe, vektorisierte Lesegeräte und optimierte Shuffles und Partitionszusammenführung.
- Aktualisierte JDBC-Treiber für alle AWS Glue nativen Quellen, einschließlich MySQL, Microsoft SQL Server, Oracle, PostgreSQL, MongoDB, und aktualisierte Spark-Bibliotheken und -Abhängigkeiten, die mit Spark 3.3.0 eingeführt wurden.
- Aktualisiert mit einem neuen Amazon-Redshift-Konnektor und JDBC-Treiber.
- Optimierter Amazon-S3-Zugriff mit aktualisiertem EMR File System (EMRFS, EMR-Dateisystem) und standardmäßig aktivierten Amazon-S3-optimierten Ausgabe-Committern.
- Optimierter Data-Catalog-Zugriff mit Partitionsindizes, Pushdown-Prädikaten, Partitionsauflistung und einem aktualisierten Hive-Metastore-Client.
- Integration mit Lake Formation für kontrollierte Katalogtabellen mit Filterung auf Zellenebene und Data-Lake-Transaktionen.
- Verringerte Startlatenz zur Verbesserung der Gesamtzeiten für die Auftragserfüllung und Interaktivität.
- Spark-Aufträge werden in 1-Sekunden-Schritten mit einer zehnmal geringeren Mindestabrechnungsdauer abgerechnet – von mindestens 10 Minuten bis zu mindestens 1 Minute.

- Native Unterstützung für Open-Data-Lake-Frameworks mit Apache Hudi, Delta Lake und Apache Iceberg.
- Native Unterstützung für das Amazon-S3-basierte Cloud-Shuffle-Speicher-Plugin (ein Apache-Spark-Plugin) zur Verwendung von Amazon S3 für Shuffling und elastische Speicherkapazität.

Wesentliche Verbesserungen von Spark 3.1.1 zu Spark 3.3.0

Berücksichtigen Sie die folgenden Verbesserungen:

- Laufzeitfilterung auf Zeilenebene (SPARK-32268).
- ANSI-Verbesserungen (SPARK-38860).
- Verbesserungen der Fehlermeldungen (SPARK-38781).
- Unterstützung komplexer Typen für den vektorisierten Parquet-Reader (SPARK-34863).
- Unterstützung für versteckte Dateimetadaten für Spark SQL (SPARK-37273).
- Stellen Sie einen Profiler für Python/Pandas UDFs (SPARK-37443) bereit.
- Führen Sie Trigger ein. AvailableNow zum Ausführen von Streaming-Abfragen wie Trigger.Once in mehreren Batches (SPARK-36533).
- Umfassendere Datasource-V2-Pushdown-Funktionen (SPARK-38788).
- Migration von log4j1 zu log4j2 (SPARK-37814).

Weitere nennenswerte Änderungen

Beachten Sie folgende Änderungen:

- Abwärtskompatible Änderungen
  - Löschen Sie Verweise auf die Python-3.6-Unterstützung in Docs und Python/docs (SPARK-36977).
  - Entfernen Sie den benannten Tuple-Hack, indem Sie das integrierte Pickle durch Cloudpickle ersetzen (SPARK-32079).
  - Erhöhen Sie die Mindestversion von Pandas auf 1.0.5 (SPARK-37465).

Aktionen zur Migration zu AWS Glue 4.0

Ändern Sie bei vorhandenen Aufträgen die `Glue version` von der vorherigen Version auf `Glue 4.0` in der Auftragskonfiguration.

- AWS Glue Wählen `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` `Glue version` Sie in Studio in.
- Wählen Sie in der API **`4.0`** im `GlueVersion`-Parameter in der `UpdateJob`-API-Operation aus.

Wählen Sie für neue Aufträge `Glue 4.0` aus, wenn Sie Aufträge erstellen.

- Wählen Sie in der Konsole `Spark 3.3, Python 3 (Glue Version 4.0) or Spark 3.3, Scala 2 (Glue Version 3.0)` in der `Glue version` aus.
- Wählen Sie in AWS Glue Studio `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` in`Glue version`.
- Wählen Sie in der API **`4.0`** im `GlueVersion`-Parameter in der `CreateJob`-API-Operation aus.

Um Spark-Ereignisprotokolle von AWS Glue 4.0 aus Version AWS Glue 2.0 oder früher anzuzeigen, starten Sie einen aktualisierten Spark-Verlaufsserver für AWS Glue 4.0 mithilfe von AWS CloudFormation oder Docker.

Checkliste für die Migration

- Sind die externen Python-Bibliotheken Ihres Auftrags von Python 2.7/3.6 abhängig?
  - Aktualisieren Sie die abhängigen Bibliotheken von Python 2.7/3.6 auf Python 3.10, da Spark 3.3.0 die Unterstützung für Python 2.7 und 3.6 vollständig entfernt hat.

Migration von AWS Glue 3.0 auf 4.0 AWS Glue

Beachten Sie die folgenden Änderungen bei der Migration:

- Alle bestehenden Jobparameter und Hauptfunktionen, die in AWS Glue 3.0 vorhanden sind, werden auch in AWS Glue 4.0 verfügbar sein.
- AWS Glue 3.0 verwendet Amazon EMR-optimiertes Spark 3.1.1 und AWS Glue 4.0 verwendet Amazon EMR-optimiertes Spark 3.3.0.

  Einige Spark-Änderungen allein erfordern möglicherweise eine Überarbeitung Ihrer Skripts, um sicherzustellen, dass auf entfernte Features nicht verwiesen wird.
- AWS Glue 4.0 beinhaltet auch ein Update für EMRFS und Hadoop. Informationen über die jeweilige Version finden Sie unter Anhang A: Nennenswerte Aktualisierungen von Abhängigkeiten.
- Das in ETL-Jobs bereitgestellte AWS SDK wurde jetzt von 1.11 auf 1.12 aktualisiert.

- Alle Python-Aufträge werden Python-Version 3.10 verwenden. Zuvor wurde Python 3.7 in AWS Glue 3.0 verwendet.

  Infolgedessen wurden einige mitgelieferte AWS Glue Pymodule out-of-the-box aktualisiert.
- Log4j wurde auf Log4j2 aktualisiert.
  - Informationen zum Log4j2-Migrationspfad finden Sie in der Log4j-Dokumentation.
  - Sie müssen stattdessen jede benutzerdefinierte log4j.properties-Datei in eine log4j2.properties-Datei mit den entsprechenden log4j2-Eigenschaften umbenennen.
- Informationen zur Migration bestimmter Konnektoren finden Sie unter Migration von Connector und JDBC-Treibern für 4.0 AWS Glue.
- Das AWS Encryption SDK wurde von 1.x auf 2.x aktualisiert. AWS Glue Jobs, die AWS Glue Sicherheitskonfigurationen verwenden, und Jobs, die von der in Runtime bereitgestellten AWS Encryption SDK-Abhängigkeit abhängig sind, sind betroffen. Weitere Informationen finden Sie in den Anweisungen zur AWS Glue Jobmigration.

  Sie können einen AWS Glue 2.0/3.0-Job problemlos auf einen AWS Glue 4.0-Job aktualisieren, da AWS Glue 2.0/3.0 bereits die Bridge-Version des AWS Encryption SDK enthält.

Weitere Informationen finden Sie in der Dokumentation zur Spark-Migration:

- Upgrade von Spark SQL 3.1 auf 3.2
- Upgrade von Spark SQL 3.2 auf 3.3

Migration von 2.0 auf 4.0 AWS GlueAWS Glue

Beachten Sie die folgenden Änderungen bei der Migration:

> **Note**
>
> Informationen zu Migrationsschritten im Zusammenhang mit AWS Glue 3.0 finden Sie unterMigration von AWS Glue 3.0 auf 4.0 AWS Glue.

- Alle bestehenden Jobparameter und Hauptfunktionen, die es in AWS Glue 2.0 gibt, werden auch in AWS Glue 4.0 verfügbar sein.

- Der EMRFS S3-optimierte Committer zum Schreiben von Parquet-Daten in Amazon S3 ist seit 3.0 standardmäßig aktiviert. AWS Glue Sie können ihn jedoch nach wie vor deaktivieren, indem Sie `--enable-s3-parquet-optimized-committer` auf `false` einstellen.
- AWS Glue 2.0 verwendet Open-Source-Spark 2.4 und AWS Glue 4.0 verwendet Amazon EMR-optimiertes Spark 3.3.0.
  - Einige Spark-Änderungen allein erfordern möglicherweise eine Überarbeitung Ihrer Skripts, um sicherzustellen, dass auf entfernte Features nicht verwiesen wird.
  - Zum Beispiel aktiviert Spark 3.3.0 UDFs Scala-untyped nicht, aber Spark 2.4 erlaubt sie.
- Das in ETL-Jobs bereitgestellte AWS SDK wurde jetzt von 1.11 auf 1.12 aktualisiert.
- AWS Glue 4.0 beinhaltet außerdem ein Update für EMRFS, aktualisierte JDBC-Treiber und zusätzliche Optimierungen für Spark selbst, bereitgestellt von. AWS Glue
- Scala wurde von 2.11 auf 2.12 aktualisiert und Scala 2.12 ist nicht abwärtskompatibel mit Scala 2.11.
- Python 3.10 ist die Standardversion, die für Python-Skripte verwendet wird, da AWS Glue 2.0 nur Python 3.7 und 2.7 verwendet hat.
  - Python 2.7 wird mit Spark 3.3.0 nicht unterstützt. Jeder Job, der Python 2 in der Jobkonfiguration anfordert, schlägt mit einem fehl IllegalArgumentException.
  - Ein neuer Mechanismus zur Installation zusätzlicher Python-Module ist seit AWS Glue 2.0 verfügbar.
- Mehrere Abhängigkeitsaktualisierungen, hervorgehoben in Anhang A: Nennenswerte Aktualisierungen von Abhängigkeiten.
- Alle zusätzlichen JAR-Dateien, die in bestehenden AWS Glue 2.0-Jobs bereitgestellt werden, können zu widersprüchlichen Abhängigkeiten führen, da es in 4.0 von 2.0 Upgrades in mehreren Abhängigkeiten gab. Mit dem Job-Parameter können Sie Klassenpfadkonflikte in AWS Glue 4.0 vermeiden. `--user-jars-first` AWS Glue
- AWS Glue 4.0 verwendet Spark 3.3. Ab Spark 3.1 gab es eine Änderung im Verhalten von loading/saving of timestamps from/to Parquet-Dateien. Weitere Informationen finden Sie unter Upgrade von Spark SQL 3.0 auf 3.1.

  Wir empfehlen, beim Lesen/Schreiben von Parquet-Daten, die Zeitstempelspalten enthalten, die folgenden Parameter einzustellen. Durch die Einstellung dieser Parameter kann das Problem mit der Kalenderinkompatibilität behoben werden, das beim Upgrade von Spark 2 auf Spark 3 sowohl für den AWS Glue Dynamic Frame als auch für den Spark Data Frame auftritt. Verwenden Sie die

Option CORRECTED, um den Datetime-Wert so zu lesen, wie er ist, und die LEGACY-Option, um die Datetime-Werte im Hinblick auf die Kalenderdifferenz während des Lesens neu zu basieren.

```
- Key: --conf
- Value: spark.sql.legacy.parquet.int96RebaseModeInRead=[CORRECTED|LEGACY] --
conf spark.sql.legacy.parquet.int96RebaseModeInWrite=[CORRECTED|LEGACY] --conf
 spark.sql.legacy.parquet.datetimeRebaseModeInRead=[CORRECTED|LEGACY]
```

- Informationen zur Migration bestimmter Konnektoren finden Sie unter Migration von Connector und JDBC-Treibern für 4.0 AWS Glue.
- Das AWS Encryption SDK wurde von 1.x auf 2.x aktualisiert. AWS Glue Jobs, die AWS Glue Sicherheitskonfigurationen verwenden, und Jobs, die von der in Runtime bereitgestellten AWS Encryption SDK-Abhängigkeit abhängig sind, sind betroffen. Sehen Sie sich diese Anweisungen zur AWS Glue Job-Migration an:
  - Sie können einen AWS Glue 2.0-Job problemlos auf einen AWS Glue 4.0-Job aktualisieren, da AWS Glue 2.0 bereits die Bridge-Version des AWS Encryption SDK enthält.

Weitere Informationen finden Sie in der Dokumentation zur Spark-Migration:

- Upgrade von Spark SQL 2.4 auf 3.0
- Upgrade von Spark SQL 3.1 auf 3.2
- Upgrade von Spark SQL 3.2 auf 3.3
- Änderungen im Datetime-Verhalten sind ab Spark 3.0 zu erwarten.

Migration von AWS Glue 1.0 auf 4.0 AWS Glue

Beachten Sie die folgenden Änderungen bei der Migration:

- AWS Glue 1.0 verwendet Open-Source-Spark 2.4 und AWS Glue 4.0 verwendet Amazon EMR-optimiertes Spark 3.3.0.
  - Einige Spark-Änderungen allein erfordern möglicherweise eine Überarbeitung Ihrer Skripts, um sicherzustellen, dass auf entfernte Features nicht verwiesen wird.
  - Zum Beispiel aktiviert Spark 3.3.0 UDFs Scala-untyped nicht, aber Spark 2.4 erlaubt sie.
- Alle Jobs in AWS Glue 4.0 werden mit deutlich verbesserten Startzeiten ausgeführt. Spark-Aufträge werden in 1-Sekunden-Schritten mit einer 10-fach niedrigeren Mindestabrechnungsdauer in Rechnung gestellt, da die Startlatenz von maximal 10 Minuten zu maximal 1 Minute reicht.

- Das Protokollierungsverhalten hat sich in AWS Glue Version 4.0 erheblich geändert. Für Spark 3.3.0 ist die Mindestanforderung Log4j2 erforderlich.
- Mehrere Abhängigkeitsaktualisierungen, hervorgehoben im Anhang.
- Scala wurde von 2.11 auf 2.12 aktualisiert und Scala 2.12 ist nicht abwärtskompatibel mit Scala 2.11.
- Python 3.10 ist auch die Standardversion für Python-Skripte, da AWS Glue 0.9 nur Python 2 verwendet hat.

  Python 2.7 wird mit Spark 3.3.0 nicht unterstützt. Jeder Job, der Python 2 in der Jobkonfiguration anfordert, schlägt mit einem fehl IllegalArgumentException.
- Ein neuer Mechanismus zur Installation zusätzlicher Python-Module über Pip ist seit AWS Glue 2.0 verfügbar. Weitere Informationen finden Sie unter Installieren zusätzlicher Python-Module mit Pip in AWS Glue 2.0+.
- AWS Glue 4.0 läuft nicht auf Apache YARN, daher gelten die YARN-Einstellungen nicht.
- AWS Glue 4.0 hat kein Hadoop Distributed File System (HDFS).
- Alle zusätzlichen JAR-Dateien, die in bestehenden AWS Glue 1.0-Jobs bereitgestellt werden, können zu widersprüchlichen Abhängigkeiten führen, da es in 4.0 Upgrades von 1.0 in mehreren Abhängigkeiten gab. Wir aktivieren AWS Glue 4.0 standardmäßig mit dem `--user-jars-first` AWS Glue Job-Parameter, um dieses Problem zu vermeiden.
- AWS Glue 4.0 unterstützt Auto Scaling. Daher ist die ExecutorAllocationManager Metrik verfügbar, wenn Auto Scaling aktiviert ist.
- In Jobs der AWS Glue Version 4.0 geben Sie die Anzahl der Worker und den Workertyp an, aber keinen`maxCapacity`.
- AWS Glue 4.0 unterstützt noch keine Transformationen für maschinelles Lernen.
- Informationen zur Migration bestimmter Konnektoren finden Sie unter Migration von Connector und JDBC-Treibern für 4.0 AWS Glue.
- Das AWS Encryption SDK wurde von 1.x auf 2.x aktualisiert. AWS Glue Jobs, die AWS Glue Sicherheitskonfigurationen verwenden, und Jobs, die von der in Runtime bereitgestellten AWS Encryption SDK-Abhängigkeit abhängig sind, sind betroffen. Informationen zur AWS Glue Job-Migration finden Sie in diesen Anweisungen.
  - Sie können einen AWS Glue 0.9/1.0-Job nicht direkt auf einen AWS Glue 4.0-Job migrieren. Dies liegt daran, dass das Encryption SDK bei einem direkten Upgrade auf Version 2.x oder höher und bei sofortiger Aktivierung aller neuen Funktionen nicht in der Lage sein wird, den in früheren Versionen des AWS Encryption SDK verschlüsselten Chiffretext zu entschlüsseln. AWS

- Für ein sicheres Upgrade empfehlen wir zunächst, zu einem AWS Glue 2.0/3.0-Job zu migrieren, der die Bridge-Version des Encryption SDK enthält. AWS Führen Sie den Job einmal aus, um die AWS Encryption SDK Bridge-Version zu verwenden.
- Nach Abschluss können Sie den AWS Glue 2.0/3.0-Job sicher auf AWS Glue 4.0 migrieren.

Weitere Informationen finden Sie in der Dokumentation zur Spark-Migration:

- Upgrade von Spark SQL 2.4 auf 3.0
- Upgrade von Spark SQL 3.0 auf 3.1
- Upgrade von Spark SQL 3.1 auf 3.2
- Upgrade von Spark SQL 3.2 auf 3.3
- Änderungen im Datetime-Verhalten sind ab Spark 3.0 zu erwarten.

Migration von AWS Glue 0.9 auf 4.0 AWS Glue

Beachten Sie die folgenden Änderungen bei der Migration:

- AWS Glue 0.9 verwendet Open-Source-Spark 2.2.1 und AWS Glue 4.0 verwendet Amazon EMR-optimiertes Spark 3.3.0.
  - Einige Spark-Änderungen allein erfordern möglicherweise eine Überarbeitung Ihrer Skripts, um sicherzustellen, dass auf entfernte Features nicht verwiesen wird.
  - Zum Beispiel aktiviert Spark 3.3.0 UDFs Scala-untyped nicht, aber Spark 2.2 erlaubt sie.
- Alle Jobs in AWS Glue 4.0 werden mit deutlich verbesserten Startzeiten ausgeführt. Spark-Aufträge werden in 1-Sekunden-Schritten mit einer 10-mal geringeren Mindestabrechnungsdauer abgerechnet, da die Startlatenz von maximal 10 Minuten auf maximal 1 Minute sinkt.
- Das Protokollierungsverhalten hat sich seit AWS Glue 4.0 erheblich geändert. Für Spark 3.3.0 ist, wie hier erwähnt, eine Mindestanforderung von Log4j2 erforderlich (https://spark.apache.org/docs/latest/core-migration-guide.html# -32-to-33). upgrading-from-core
- Mehrere Abhängigkeitsaktualisierungen, hervorgehoben im Anhang.
- Scala wurde von 2.11 auf 2.12 aktualisiert und Scala 2.12 ist nicht abwärtskompatibel mit Scala 2.11.
- Python 3.10 ist auch die Standardversion für Python-Skripte, da AWS Glue 0.9 nur Python 2 verwendet hat.

  - Python 2.7 wird mit Spark 3.3.0 nicht unterstützt. Jeder Job, der Python 2 in der Jobkonfiguration anfordert, schlägt mit einem fehl IllegalArgumentException.
  - Ein neuer Mechanismus zum Installieren zusätzlicher Python-Module über Pip ist verfügbar.
- AWS Glue 4.0 läuft nicht auf Apache YARN, daher gelten die YARN-Einstellungen nicht.
- AWS Glue 4.0 hat kein Hadoop Distributed File System (HDFS).
- Alle zusätzlichen JAR-Dateien, die in bestehenden AWS Glue 0.9-Aufträgen bereitgestellt werden, können zu widersprüchlichen Abhängigkeiten führen, da es in 3.0 Upgrades von 0.9 in mehreren Abhängigkeiten gab. Mit dem Job-Parameter können Sie Klassenpfadkonflikte in AWS Glue 3.0 vermeiden. `--user-jars-first` AWS Glue
- AWS Glue 4.0 unterstützt Auto Scaling. Daher ist die ExecutorAllocationManager Metrik verfügbar, wenn Auto Scaling aktiviert ist.
- In Jobs der AWS Glue Version 4.0 geben Sie die Anzahl der Worker und den Workertyp an, aber keinen`maxCapacity`.
- AWS Glue 4.0 unterstützt noch keine Transformationen für maschinelles Lernen.
- Informationen zur Migration bestimmter Konnektoren finden Sie unter Migration von Connector und JDBC-Treibern für 4.0 AWS Glue.
- Das AWS Encryption SDK wurde von 1.x auf 2.x aktualisiert. AWS Glue Jobs, die AWS Glue Sicherheitskonfigurationen verwenden, und Jobs, die von der in Runtime bereitgestellten AWS Encryption SDK-Abhängigkeit abhängig sind, sind betroffen. Informationen zur AWS Glue Job-Migration finden Sie in diesen Anweisungen.
  - Sie können einen AWS Glue 0.9/1.0-Job nicht direkt auf einen AWS Glue 4.0-Job migrieren. Dies liegt daran, dass das Encryption SDK bei einem direkten Upgrade auf Version 2.x oder höher und bei sofortiger Aktivierung aller neuen Funktionen nicht in der Lage sein wird, den in früheren Versionen des AWS Encryption SDK verschlüsselten Chiffretext zu entschlüsseln. AWS
  - Für ein sicheres Upgrade empfehlen wir zunächst, zu einem AWS Glue 2.0/3.0-Job zu migrieren, der die Bridge-Version des Encryption SDK enthält. AWS Führen Sie den Job einmal aus, um die AWS Encryption SDK Bridge-Version zu verwenden.
  - Nach Abschluss können Sie den AWS Glue 2.0/3.0-Job sicher auf AWS Glue 4.0 migrieren.

Weitere Informationen finden Sie in der Dokumentation zur Spark-Migration:

- Upgrade von Spark SQL 2.2 auf 2.3
- Upgrade von Spark SQL 2.3 auf 2.4

- Upgrade von Spark SQL 2.4 auf 3.0
- Upgrade von Spark SQL 3.0 auf 3.1
- Upgrade von Spark SQL 3.1 auf 3.2
- Upgrade von Spark SQL 3.2 auf 3.3
- Änderungen im Datetime-Verhalten sind ab Spark 3.0 zu erwarten.

Migration von Connector und JDBC-Treibern für 4.0 AWS Glue

Die aktualisierten Versionen von JDBC- und Data-Lake-Konnektoren finden Sie unter:

- Anhang B: Aktualisierungen von JDBC-Treibern
- Anhang C: Konnektor-Upgrades

Hudi

- Verbesserungen der Spark-SQL-Unterstützung:
  - Durch den `Call Procedure`-Befehl wird Unterstützung für Upgrade, Downgrade, Bootstrap, Bereinigung und Reparatur hinzugefügt. Die `Create/Drop/Show/Refresh Index`-Syntax ist in Spark SQL möglich.
  - Eine Leistungslücke zwischen der Nutzung über Spark und über DataSource Spark-SQL wurde geschlossen. Datenquellenschreibvorgänge waren in der Vergangenheit schneller als SQL.
  - Alle integrierten Schlüsselgeneratoren implementieren leistungsfähigere Spark-spezifische API-Operationen.
  - Die UDF-Transformation im `insert` Massenvorgang wurde durch RDD-Transformationen ersetzt, um die Nutzungskosten zu senken. SerDe
  - Spark SQL mit Hudi erfordert die Angabe eines `primaryKey` durch `tblproperites` oder Optionen in der SQL-Anweisung. Für Aktualisierungs- und Löschvorgänge ist das `preCombineField` ebenfalls erforderlich.
- Jede Hudi-Tabelle, die vor Version 0.10.0 ohne ein `primaryKey` erstellt wurde, muss seit Version 0.10.0 mit einem `primaryKey`-Feld neu erstellt werden.

PostgreSQL

- Verschiedene Sicherheitslücken (CVEs) wurden behoben.

- Java 8 wird nativ unterstützt.
- Wenn der Auftrag mit Ausnahme von Byte-Arrays Arrays von Arrays verwendet, kann dieses Szenario als mehrdimensionale Arrays behandelt werden.

MongoDB

- Der aktuelle MongoDB-Konnektor unterstützt Spark-Version 3.1 oder höher und MongoDB-Version 4.0 oder höher.
- Aufgrund des Konnektor-Upgrades haben sich einige Eigenschaftsnamen geändert. Beispielsweise wurde der URI-Eigenschaftsname in `connection.uri` geändert. Weitere Informationen zu den aktuellen Optionen finden Sie im Blog zu MongoDB-Spark-Konnektor.
- Die Verwendung von MongoDB 4.0, das von Amazon DocumentDB gehostet wird, weist einige funktionale Unterschiede auf. Weitere Informationen enthalten die folgenden Themen:
  - Funktionale Unterschiede: Amazon DocumentDB und MongoDB
  - Unterstützte MongoDB APIs, Operationen und Datentypen.
- Die Option „Partitionierer“ ist auf `ShardedPartitioner`, `PaginateIntoPartitionsPartitioner`, und `SinglePartitionPartitioner` beschränkt. Es kann die Standardwerte `SamplePartitioner` und `PaginateBySizePartitioner` für Amazon DocumentDB nicht verwenden, da der Stage-Operator die MongoDB-API nicht unterstützt. Weitere Informationen finden Sie unter Unterstützte MongoDB APIs, Operationen und Datentypen.

Delta Lake

- Delta Lake unterstützt jetzt Zeitreisen in SQL zum einfachen Abfragen älterer Daten. Mit diesem Update sind Zeitreisen jetzt sowohl in Spark SQL als auch über die DataFrame API verfügbar. Für die aktuelle Version von TIMESTAMP in SQL wurde Unterstützung hinzugefügt.
- Spark 3.3 führt Trigger ein. AvailableNowzum Ausführen von Streaming-Abfragen als Äquivalent `Trigger.Once` zu Batch-Abfragen. Diese Unterstützung ist auch bei Verwendung von Delta-Tabellen als Streaming-Quelle verfügbar.
- Unterstützung für SHOW COLUMNS, um die Liste der Spalten in einer Tabelle zurückzugeben.
- Support für DESCRIBE DETAIL in der Scala- und DeltaTable Python-API. Es ruft mithilfe der DeltaTable API oder Spark SQL detaillierte Informationen über eine Delta-Tabelle ab.
- Unterstützung für die Rückgabe von Vorgangsmetriken aus SQL-Befehlen zum Löschen, Zusammenführen und Aktualisieren. Bisher gaben diese SQL-Befehle einen leeren Wert zurück

DataFrame, jetzt geben sie ein DataFrame mit nützlichen Metriken über den ausgeführten Vorgang zurück.

- Leistungsverbesserungen optimieren:
  - Legen Sie die Konfigurationsoption `spark.databricks.delta.optimize.repartition.enabled=true` so fest, dass im Befehl Optimieren `repartition(1)` anstelle von `coalesce(1)` verwendet wird, um eine bessere Leistung beim Komprimieren vieler kleiner Dateien zu erzielen.
  - Verbesserte Leistung durch Verwendung eines warteschlangenbasierten Ansatzes zur Parallelisierung von Verdichtungsaufträgen.
- Weitere nennenswerte Änderungen:
  - Unterstützung für die Verwendung von Variablen in den VACUUM- und OPTIMIZE-SQL-Befehlen.
  - Verbesserungen für CONVERT TO DELTA mit Katalogtabellen, einschließlich:
    - Füllen Sie das Partitionsschema automatisch aus dem Katalog aus, wenn es nicht bereitgestellt wird.
    - Verwenden Sie die Partitionsinformationen aus dem Katalog, um die festzuschreibenden Datendateien zu finden, anstatt einen vollständigen Verzeichnisscan durchzuführen. Anstatt alle Datendateien im Tabellenverzeichnis festzuschreiben, werden nur Datendateien in den Verzeichnissen der aktiven Partitionen festgeschrieben.
  - Unterstützung für CDF-Batchlesevorgänge (Change Data Feed) in Tabellen mit aktivierter Spaltenzuordnung, wenn DROP COLUMN und RENAME COLUMN nicht verwendet wurden. Weitere Informationen finden Sie in der Delta-Lake-Dokumentation.
  - Verbessern Sie die Leistung des Aktualisierungs-Befehls, indem Sie die Schemabereinigung im ersten Durchgang aktivieren.

Apache Iceberg

- Es wurden mehrere Leistungsverbesserungen für die Scan-Planung und Spark-Abfragen hinzugefügt.
- Es wurde ein allgemeiner REST-Katalog-Client hinzugefügt, der änderungsbasierte Commits verwendet, um Commit-Konflikte innerhalb des Services zu lösen.
- `AS OF`-Syntax für SQL-Zeitreiseabfragen wird unterstützt.
- merge-on-readUnterstützung für MERGE- und UPDATE-Abfragen hinzugefügt.
- Es wurde Unterstützung zum Neuschreiben von Partitionen mit Z-Reihenfolge hinzugefügt.

- Es wurde eine Spezifikation und Implementierung für Puffin hinzugefügt, ein Format für große Statistik- und Index-Blobs, wie Theta-Skizzen oder Bloom-Filter.
- Es wurden neue Schnittstellen für die inkrementelle Nutzung von Daten hinzugefügt (sowohl Append- als auch Changelog-Scans).
- Es wurde Unterstützung für Massenvorgänge und Bereichslesevorgänge für FileIO-Schnittstellen hinzugefügt.
- Es wurden weitere Metadaten-Tabellen hinzugefügt, um gelöschte Dateien in der Metadaten-Struktur anzuzeigen.
- Das Verhalten der Ablagetabelle hat sich geändert. In Iceberg 0.13.1 entfernt das Ausführen von `DROP TABLE` die Tabelle aus dem Katalog und löscht auch den Tabelleninhalt. In Iceberg 1.0.0 entfernt `DROP TABLE` nur die Tabelle aus dem Katalog. Um den Tabelleninhalt zu löschen, verwenden Sie `DROP TABLE PURGE`.
- In Iceberg 1.0.0 sind vektorisierte Parquet-Lesevorgänge standardmäßig aktiviert. Wenn Sie vektorisierte Lesevorgänge deaktivieren möchten, setzen Sie `read.parquet.vectorization.enabled` auf `false`.

Oracle

Die Änderungen sind gering.

MySQL

Die Änderungen sind gering.

Amazon Redshift

AWS Glue 4.0 bietet einen neuen Amazon Redshift Redshift-Connector mit einem neuen JDBC-Treiber. Informationen zu den Verbesserungen und zur Migration von früheren AWS Glue Versionen finden Sie unter. the section called “Redshift-Verbindungen”

Anhang A: Nennenswerte Aktualisierungen von Abhängigkeiten

Im Folgenden sind Abhängigkeits-Upgrades aufgeführt:

-Abhängigkeit	Version in AWS Glue 4.0	Version in AWS Glue 3.0	Version in AWS Glue 2.0	Version in AWS Glue 1.0
Spark	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3

-Abhängigkeit	Version in AWS Glue 4.0	Version in AWS Glue 3.0	Version in AWS Glue 2.0	Version in AWS Glue 1.0
Hadoop	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Scala	2.12	2.12	2.11	2.11
Jackson	2.13.3	2.10.x	2.7.x	2.7.x
Hive	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2,54,0	2,46,0	2.38.0	2.30.0
JSON4s	3.7.0-M11	3.6.6	3.5.x	3.5.x
Arrow	7.0.0	2.0.0	0.10.0	0.10.0
AWS Glue Datenkatalog-Client	3.7.0	3.0.0	1.10.0	N/A
Python	3.10	3.7	2.7 und 3.6	2.7 und 3.6
Boto	1,26	1,18	1.12	N/A

Anhang B: Aktualisierungen von JDBC-Treibern

Die folgenden JDBC-Treiber-Upgrades sind:

Treiber	JDBC-Treiberversion in früheren Versionen AWS Glue	JDBC-Treiberversion in 3.0 AWS Glue	JDBC-Treiberversion in 4.0 AWS Glue
MySQL	5.1	8.0.23	8.0.23
Microsoft SQL Server	6.1.0	7.0.0	9.4.0
Oracle-Datenbanken	11.2	21,1	21,7
PostgreSQL	42,10	42,2,18	42,3,6

Treiber	JDBC-Treiberversion in früheren Versionen AWS Glue	JDBC-Treiberversion in 3.0 AWS Glue	JDBC-Treiberversion in 4.0 AWS Glue
MongoDB	2.0.0	4.0.0	4.7.2
Amazon Redshift	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017	redshift-jdbc42-2.1.0.16

Anhang C: Konnektor-Upgrades

Im Folgenden sind Konnektor-Upgrades aufgeführt:

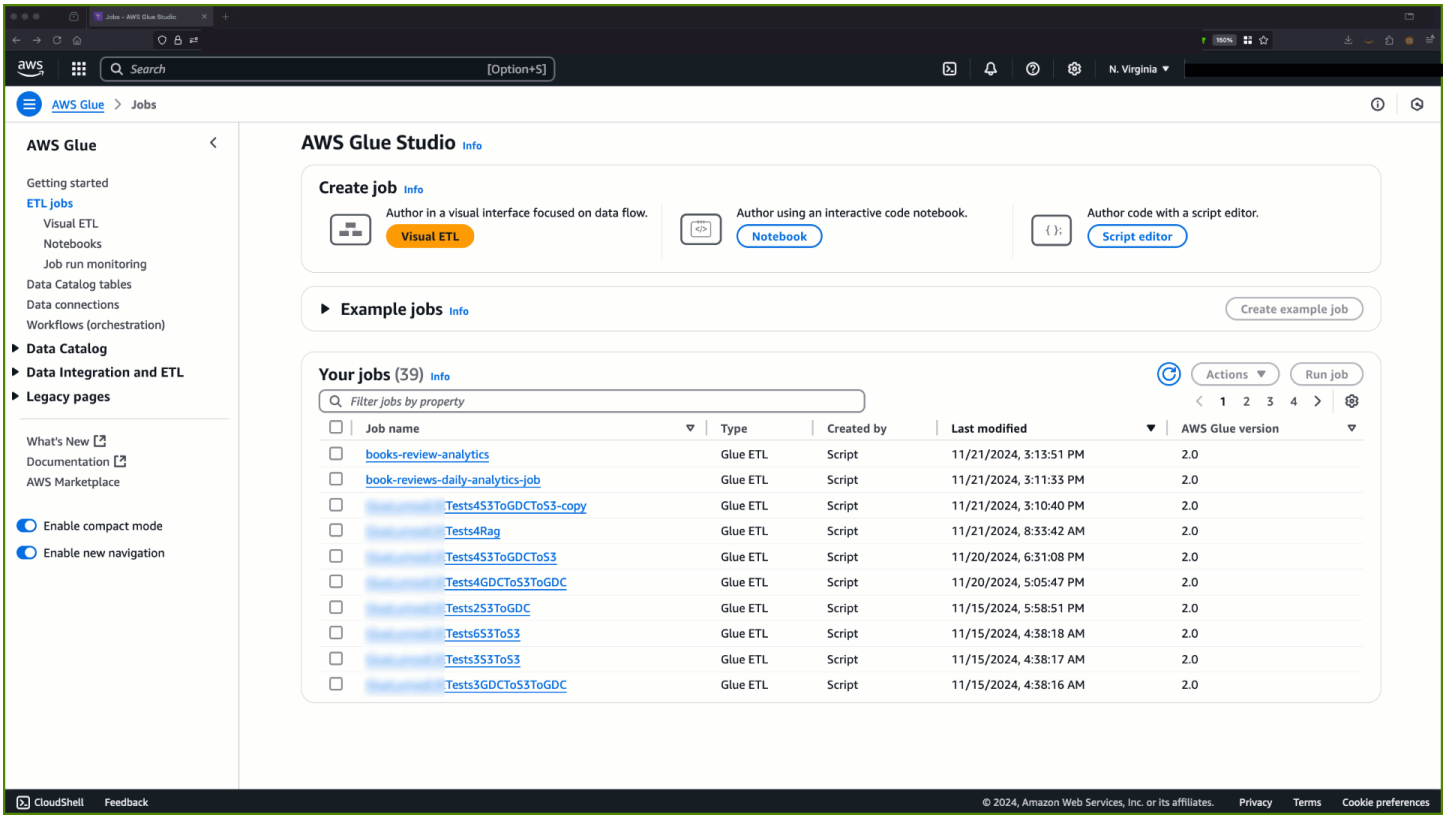
Treiber	Connector-Version in 3.0 AWS Glue	Connector-Version in AWS Glue 4.0
MongoDB	3.0.0	10.0.4
Hudi	0.10.1	0.12.1
Delta Lake	1.0.0	2.1.0
Iceberg	0.13.1	1.0.0
DynamoDB	1.11	1.12

Generative KI-Upgrades für Apache Spark in AWS Glue

Die generativen KI-Upgrades für Apache Spark Preview sind für die AWS Glue-Versionen 4.0 und 5.0 in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Tokio) und Asien-Pazifik (Sydney). Die Vorschaufunktionen können sich ändern.

Mit Spark-Upgrades in AWS Glue können Dateningenieure und Entwickler ihre bestehenden AWS Glue Spark-Jobs mithilfe generativer KI auf die neuesten Spark-Versionen aktualisieren und migrieren. Dateningenieure können damit ihre AWS Glue Spark-Jobs scannen, Upgrade-Pläne

erstellen, Pläne ausführen und Ergebnisse validieren. Es reduziert den Zeit- und Kostenaufwand für Spark-Upgrades, indem es die undifferenzierte Arbeit der Identifizierung und Aktualisierung von Spark-Skripten, Konfigurationen, Abhängigkeiten, Methoden und Funktionen automatisiert.



Funktionsweise

Wenn Sie die Upgrade-Analyse verwenden, identifiziert AWS Glue Unterschiede zwischen Versionen und Konfigurationen im Code Ihres Jobs, um einen Upgrade-Plan zu generieren. Der Upgrade-Plan beschreibt alle Codeänderungen und die erforderlichen Migrationsschritte. Als Nächstes erstellt AWS Glue die aktualisierte Anwendung und führt sie in einer Umgebung aus, um Änderungen zu validieren, und generiert eine Liste mit Codeänderungen, damit Sie Ihren Job migrieren können. Sie können sich das aktualisierte Skript zusammen mit der Zusammenfassung ansehen, in der die vorgeschlagenen Änderungen detailliert beschrieben werden. Nachdem Sie Ihre eigenen Tests ausgeführt haben, akzeptieren Sie die Änderungen und der AWS Glue-Job wird automatisch mit dem neuen Skript auf die neueste Version aktualisiert.

Der Upgrade-Analyseprozess kann je nach Komplexität des Jobs und Arbeitslast einige Zeit in Anspruch nehmen. Die Ergebnisse der Upgrade-Analyse werden im angegebenen Amazon S3 S3-Pfad gespeichert, der überprüft werden kann, um das Upgrade und mögliche Kompatibilitätsprobleme zu verstehen. Nachdem Sie die Ergebnisse der Upgrade-Analyse überprüft haben, können

Sie entscheiden, ob Sie mit dem eigentlichen Upgrade fortfahren oder vor dem Upgrade die erforderlichen Änderungen am Job vornehmen möchten.

Voraussetzungen

Die folgenden Voraussetzungen sind erforderlich, um generative KI zum Upgrade von Jobs in AWS Glue zu verwenden:

- AWS Glue 2 PySpark Jobs — Nur AWS Glue 2-Jobs können auf AWS Glue 5 aktualisiert werden.
- IAM-Berechtigungen sind erforderlich, um die Analyse zu starten, die Ergebnisse zu überprüfen und Ihren Job zu aktualisieren. Weitere Informationen finden Sie in den Beispielen im folgenden Berechtigungen Abschnitt.
- Wenn AWS KMS Sie Analyseartefakte verschlüsseln möchten, sind zusätzliche AWS AWS KMS Berechtigungen erforderlich. Weitere Informationen finden Sie in den Beispielen im folgenden AWS KMS Richtlinie Abschnitt.

Berechtigungen

Um eine neue Upgrade-Analyse zu starten, benötigen Sie die folgenden Berechtigungen:

1. Aktualisieren Sie die IAM-Richtlinie des Aufrufers mit der folgenden Berechtigung:

```
{
"Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["glue:StartJobUpgradeAnalysis",
                 "glue:StartJobRun",
                 "glue:GetJobRun",
                 "glue:GetJob",
                 "glue:BatchStopJobRun"
                 ],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:job/jobName"
      ]
    },
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": [
```

```
        "<s3 script location associated with the job>"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
            "s3:PutObject",
            "s3:GetObject"

        ],
      "Resource": [
        "<result s3 path provided on API>"
      ]
    },
    {
    "Effect": "Allow",
     "Action": [
       "kms:Decrypt",
       "kms:GenerateDataKey",
      ],
     "Resource": "<key-arn-passed in the API>"
    }
  ]
}
```

2. Aktualisieren Sie die Ausführungsrolle des Auftrags, den Sie aktualisieren, sodass sie die folgende Inline-Richtlinie enthält:

```
    {
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": [
        "ARN of the Amazon S3 path provided on API",
        "ARN of the Amazon S3 path provided on API/*"
      ]
    }
```

Wenn Sie beispielsweise den Amazon S3-Pfad verwenden`s3://amzn-s3-demo-bucket/upgraded-result`, lautet die Richtlinie:

```
{
      "Effect": "Allow",
```

```
    "Action": ["s3:GetObject"],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket/upgraded-result/",
      "arn:aws:s3:::amzn-s3-demo-bucket/upgraded-result/*"
    ]
  }
```

Um die Details einer Analyse abzurufen, benötigen Sie die folgenden Berechtigungen:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["glue:GetJobUpgradeAnalysis"],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:job/jobName"
      ]
    }
  ]
}
```

Um eine laufende Analyse zu beenden, benötigen Sie die folgenden Berechtigungen:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["glue:StopJobUpgradeAnalysis",
                 "glue:BatchStopJobRun"
                 ],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:job/jobName"
      ]
```

```
    }
  ]
}
```

Um alle Analysen aufzulisten, die für einen bestimmten Job eingereicht wurden, benötigen Sie die folgenden Berechtigungen:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["glue:ListJobUpgradeAnalyses"],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:job/jobName"
      ]
    }
  ]
}
```

Um die Änderungen aus einer Analyse zu akzeptieren und einen Job zu aktualisieren, benötigen Sie die folgenden Berechtigungen:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["glue:UpdateJob",
                 "glue:UpgradeJob"
                 ],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:job/jobName"
      ]
    },
```

```
    {
      "Effect": "Allow",
      "Action": ["iam:PassRole"],
      "Resource": [
        "<Role arn associated with the job>"
      ]
    }
  ]
}
```

AWS KMS Richtlinie

Um beim Starten einer Analyse Ihren eigenen benutzerdefinierten AWS KMS Schlüssel zu übergeben, lesen Sie bitte den folgenden Abschnitt, um die entsprechenden Berechtigungen für die AWS KMS Schlüssel zu konfigurieren.

Konfiguration der Verschlüsselung der Ergebnisartefakte mithilfe eines AWS KMS Schlüssels:

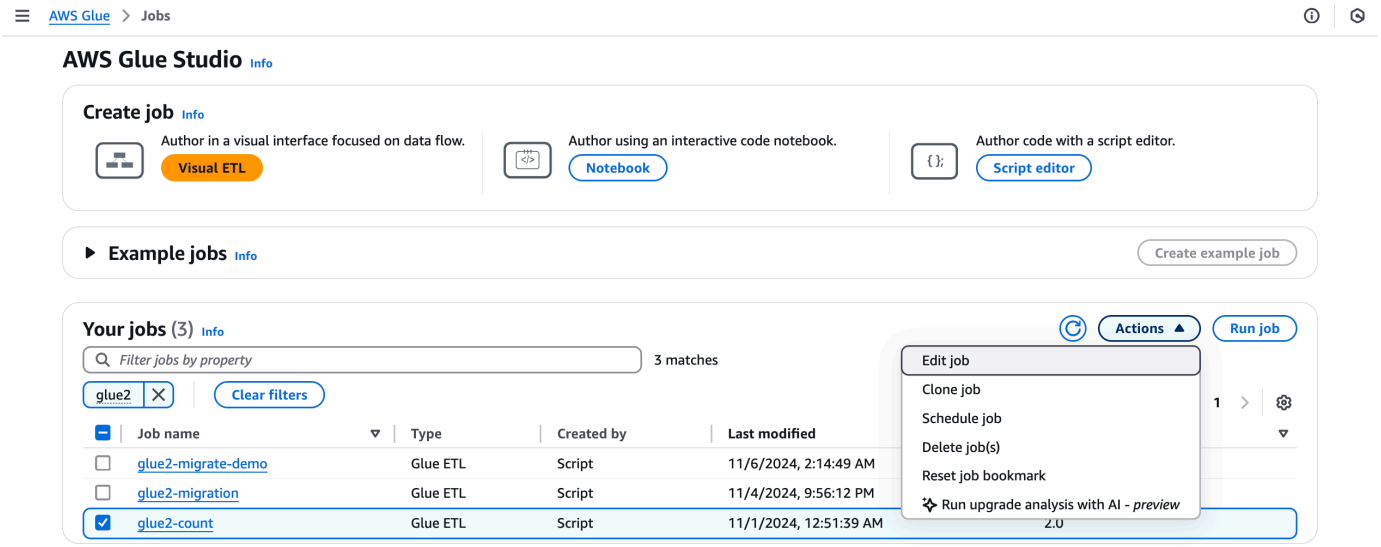
Diese Richtlinie stellt sicher, dass Sie sowohl über die Verschlüsselungs- als auch über die Entschlüsselungsberechtigungen für den AWS KMS Schlüssel verfügen.

```
{
    "Effect": "Allow",
    "Principal":{
        "AWS": "<IAM Customer caller ARN>"
    },
    "Action": [
      "kms:Decrypt",
      "kms:GenerateDataKey",
    ],
    "Resource": "<key-arn-passed-on-start-api>"
}
```

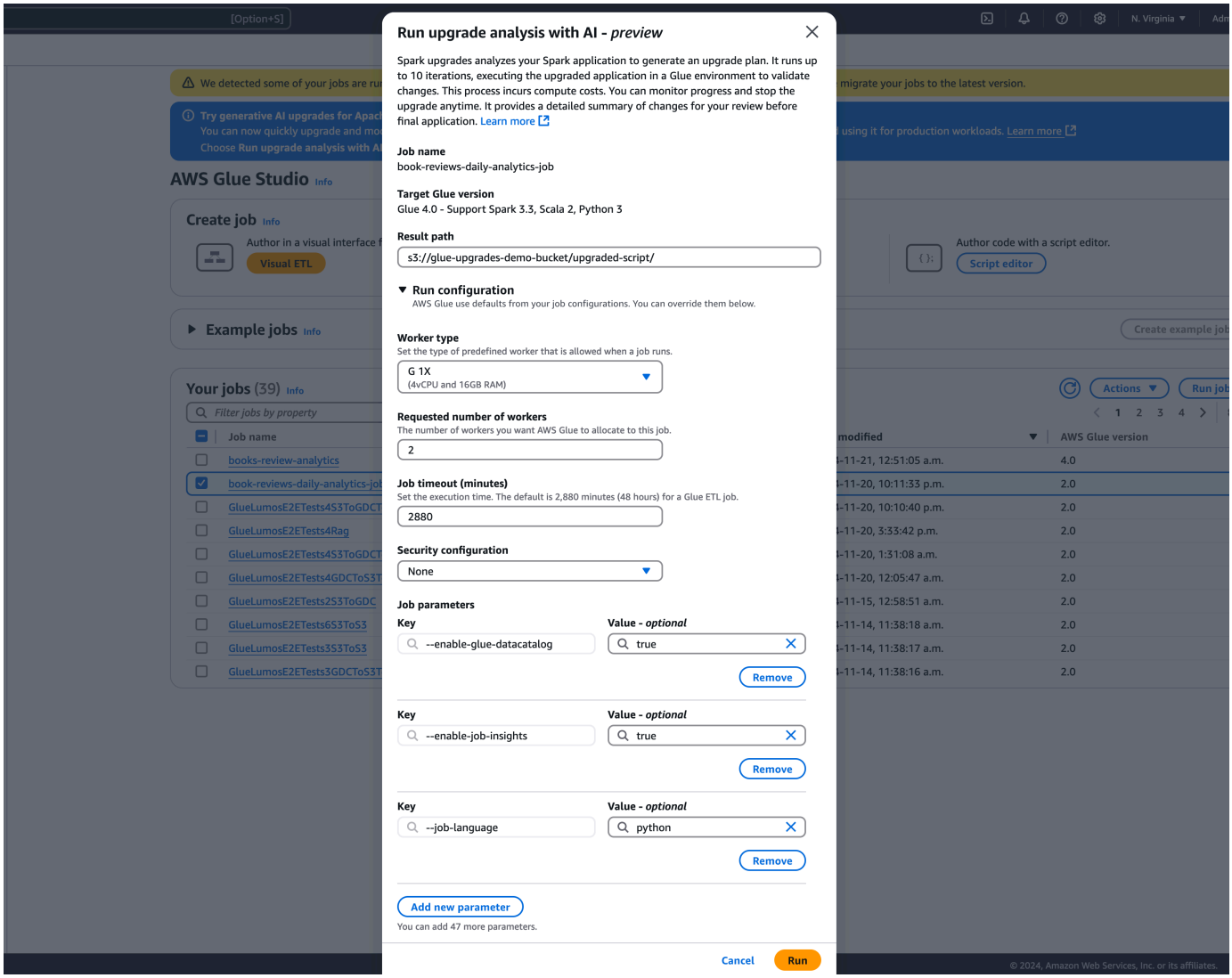
Ausführen einer Upgrade-Analyse und Anwenden des Upgrade-Skripts

Sie können eine Upgrade-Analyse ausführen, die einen Upgrade-Plan für einen Job generiert, den Sie in der Ansicht Jobs auswählen.

1. Wählen Sie unter Jobs einen AWS Glue 2.0-Job aus und wählen Sie dann im Menü Aktionen die Option Upgrade-Analyse ausführen aus.



2. Wählen Sie im Modal einen Pfad aus, um Ihren generierten Upgrade-Plan im Ergebnispfad zu speichern. Dies muss ein Amazon S3 S3-Bucket sein, auf den Sie zugreifen und in den Sie schreiben können.



3. Konfigurieren Sie bei Bedarf zusätzliche Optionen:

- Konfiguration ausführen — optional: Die Ausführungskonfiguration ist eine optionale Einstellung, mit der Sie verschiedene Aspekte der während der Upgrade-Analyse durchgeführten Validierungsläufe anpassen können. Diese Konfiguration wird zur Ausführung des aktualisierten Skripts verwendet und ermöglicht Ihnen die Auswahl der Eigenschaften der Rechenumgebung (Workertyp, Anzahl der Worker usw.). Beachten Sie, dass Sie Ihre Entwicklerkonten außerhalb der Produktionsumgebung verwenden sollten, um die Validierungen von Beispieldatensätzen durchzuführen, bevor Sie die Änderungen überprüfen, akzeptieren und auf Produktionsumgebungen anwenden. Die Ausführungskonfiguration umfasst die folgenden anpassbaren Parameter:

- Workertyp: Sie können den Workertyp angeben, der für die Validierungsläufe verwendet werden soll, sodass Sie je nach Ihren Anforderungen die geeigneten Rechenressourcen auswählen können.
- Anzahl der Mitarbeiter: Sie können die Anzahl der Mitarbeiter definieren, die für die Validierungsläufe bereitgestellt werden sollen, sodass Sie die Ressourcen entsprechend Ihren Workload-Anforderungen skalieren können.
- Job-Timeout (in Minuten): Mit diesem Parameter können Sie ein Zeitlimit für die Validierungsläufe festlegen und so sicherstellen, dass die Jobs nach einer bestimmten Dauer beendet werden, um einen übermäßigen Ressourcenverbrauch zu vermeiden.
- Sicherheitskonfiguration: Sie können Sicherheitseinstellungen wie Verschlüsselung und Zugriffskontrolle konfigurieren, um den Schutz Ihrer Daten und Ressourcen während der Validierungsläufe zu gewährleisten.
- Zusätzliche Job-Parameter: Bei Bedarf können Sie neue Job-Parameter hinzufügen, um die Ausführungsumgebung für die Validierungsläufe weiter anzupassen.

Durch die Nutzung der Ausführungskonfiguration können Sie die Validierungsläufe an Ihre spezifischen Anforderungen anpassen. Sie können die Validierungsläufe beispielsweise so konfigurieren, dass ein kleinerer Datensatz verwendet wird, wodurch die Analyse schneller abgeschlossen werden kann und die Kosten optimiert werden. Dieser Ansatz stellt sicher, dass die Upgrade-Analyse effizient durchgeführt wird und gleichzeitig die Ressourcenauslastung und die damit verbundenen Kosten während der Validierungsphase minimiert werden.

- Verschlüsselungskonfiguration — optional:
  - Verschlüsselung von Upgrade-Artefakten aktivieren: Aktiviert die Verschlüsselung im Ruhezustand, wenn Daten in den Ergebnispfad geschrieben werden. Wenn Sie Ihre Upgrade-Artefakte nicht verschlüsseln möchten, lassen Sie diese Option deaktiviert.

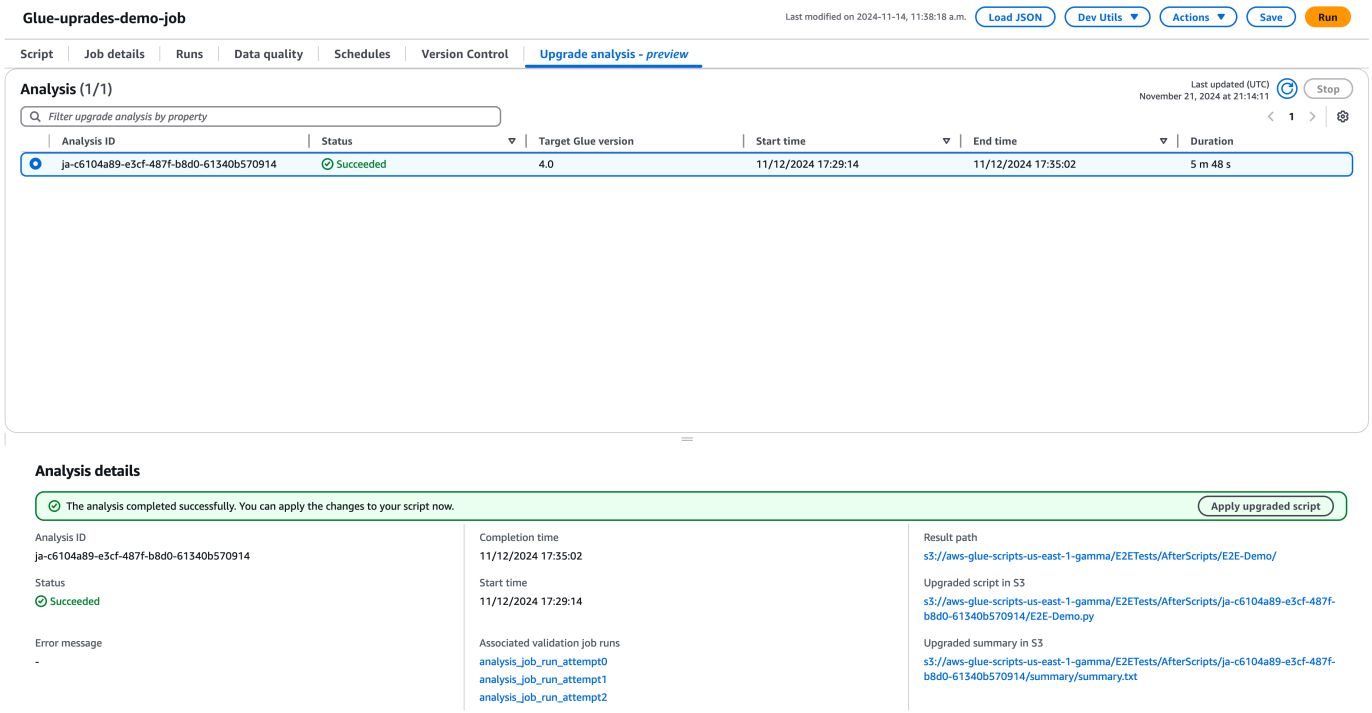
4. Wählen Sie Ausführen, um die Upgrade-Analyse zu starten. Während die Analyse ausgeführt wird, können Sie die Ergebnisse auf der Registerkarte Upgrade-Analyse anzeigen. Im Fenster mit den Analysedetails werden Informationen zur Analyse sowie Links zu den Upgrade-Artefakten angezeigt.

   - Ergebnispfad — Hier werden die Zusammenfassung der Ergebnisse und das Upgrade-Skript gespeichert.
   - Aktualisiertes Skript in Amazon S3 — der Speicherort des Upgrade-Skripts in Amazon S3. Sie können sich das Skript ansehen, bevor Sie das Upgrade anwenden.

- Upgrade-Zusammenfassung in Amazon S3 — der Speicherort der Upgrade-Zusammenfassung in Amazon S3. Sie können sich die Upgrade-Zusammenfassung ansehen, bevor Sie das Upgrade durchführen.

5. Wenn die Upgrade-Analyse erfolgreich abgeschlossen wurde, können Sie das Upgrade-Skript anwenden, um Ihren Job automatisch zu aktualisieren, indem Sie Upgrade-Skript anwenden wählen.

   Nach der Anwendung wird die AWS Glue-Version auf 4.0 aktualisiert. Sie können das neue Skript auf der Registerkarte Skript anzeigen.



Verstehen Sie Ihre Upgrade-Zusammenfassung

Dieses Beispiel zeigt, wie ein AWS Glue-Job von Version 2.0 auf Version 4.0 aktualisiert wird. Der Beispieljob liest Produktdaten aus einem Amazon S3 S3-Bucket, wendet mithilfe von Spark SQL mehrere Transformationen auf die Daten an und speichert dann die transformierten Ergebnisse wieder in einem Amazon S3 S3-Bucket.

Originalcode (AWS Glue 2.0) - vor dem Upgrade

```
from awsglue.transforms import *
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
from awsglue.job import Job
import json
from pyspark.sql.types import StructType

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)

gdc_database = "s3://aws-glue-scripts-us-east-1-gamma/demo-database/"
schema_location = (
    "s3://aws-glue-scripts-us-east-1-gamma/DataFiles/"
)

products_schema_string = spark.read.text(
    f"{schema_location}schemas/products_schema"
).first()[0]

product_schema = StructType.fromJson(json.loads(products_schema_string))

products_source_df = (
    spark.read.option("header", "true")
    .schema(product_schema)
    .option(
        "path",
        f"{gdc_database}products/",
    )
    .csv(f"{gdc_database}products/")
)

products_source_df.show()
products_temp_view_name = "spark_upgrade_demo_product_view"
products_source_df.createOrReplaceTempView(products_temp_view_name)

query = f"select {products_temp_view_name}.*, format_string('%0$s-%0$s', category,
 subcategory) as unique_category from {products_temp_view_name}"
products_with_combination_df = spark.sql(query)
products_with_combination_df.show()

products_with_combination_df.createOrReplaceTempView(products_temp_view_name)
product_df_attribution = spark.sql(
```

```
    f"""
SELECT *,
unbase64(split(product_name, ' ')[0]) as product_name_decoded,
unbase64(split(unique_category, '-')[1]) as subcategory_decoded
FROM {products_temp_view_name}
"""
)
product_df_attribution.show()


product_df_attribution.write.mode("overwrite").option("header", "true").option(
    "path", f"{gdc_database}spark_upgrade_demo_product_agg/"
).saveAsTable("spark_upgrade_demo_product_agg", external=True)

spark_upgrade_demo_product_agg_table_df = spark.sql(
    f"SHOW TABLE EXTENDED in default like 'spark_upgrade_demo_product_agg'"
)
spark_upgrade_demo_product_agg_table_df.show()
job.commit()
```

Neuer Code (Glue 4.0) - nach dem Upgrade

```
from awsglue.transforms import *
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
from awsglue.job import Job
import json
from pyspark.sql.types import StructType

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
# change 1
spark.conf.set("spark.sql.adaptive.enabled", "false")
# change 2
spark.conf.set("spark.sql.legacy.pathOptionBehavior.enabled", "true")
job = Job(glueContext)

gdc_database = "s3://aws-glue-scripts-us-east-1-gamma/demo-database/"
schema_location = (
    "s3://aws-glue-scripts-us-east-1-gamma/DataFiles/"
```

```
)

products_schema_string = spark.read.text(
    f"{schema_location}schemas/products_schema"
).first()[0]

product_schema = StructType.fromJson(json.loads(products_schema_string))

products_source_df = (
    spark.read.option("header", "true")
    .schema(product_schema)
    .option(
        "path",
        f"{gdc_database}products/",
    )
    .csv(f"{gdc_database}products/")
)

products_source_df.show()
products_temp_view_name = "spark_upgrade_demo_product_view"
products_source_df.createOrReplaceTempView(products_temp_view_name)

# change 3
query = f"select {products_temp_view_name}.*, format_string('%1$s-%1$s', category,
 subcategory) as unique_category from {products_temp_view_name}"
products_with_combination_df = spark.sql(query)
products_with_combination_df.show()

products_with_combination_df.createOrReplaceTempView(products_temp_view_name)
# change 4
product_df_attribution = spark.sql(
    f"""
SELECT *,
try_to_binary(split(product_name, ' ')[0], 'base64') as product_name_decoded,
try_to_binary(split(unique_category, '-')[1], 'base64') as subcategory_decoded
FROM {products_temp_view_name}
"""
)
product_df_attribution.show()


product_df_attribution.write.mode("overwrite").option("header", "true").option(
    "path", f"{gdc_database}spark_upgrade_demo_product_agg/"
).saveAsTable("spark_upgrade_demo_product_agg", external=True)
```

```
spark_upgrade_demo_product_agg_table_df = spark.sql(
    f"SHOW TABLE EXTENDED in default like 'spark_upgrade_demo_product_agg'"
)
spark_upgrade_demo_product_agg_table_df.show()
job.commit()
```

Erläuterung der Zusammenfassung der Analyse

```
In Spark 3.2, spark.sql.adaptive.enabled is enabled by default. To restore the behavior before
Spark 3.2,
you can set spark.sql.adaptive.enabled to false.


In Spark 3.1, path option cannot coexist when the following methods are called with path
parameter(s): DataFrameReader.load(), DataFrameWriter.save(), DataStreamReader.load(), or
DataStreamWriter.start(). In addition, paths option cannot coexist for DataFrameReader.load().
For example, spark.read.format(csv).option(path, /tmp).load(/tmp2) or spark.read.option(path, /
tmp).csv(/tmp2) will throw org.apache.spark.sql.AnalysisException. In Spark version 3.0 and
below, path option is overwritten if one path parameter is passed to above methods; path option
is added to the overall paths if multiple path parameters are passed to DataFrameReader.load().
To restore the behavior before Spark 3.1, you can set
spark.sql.legacy.pathOptionBehavior.enabled to true.


Since Spark 3.3, the `strfmt` in `format_string(strfmt, obj, ...)` and `printf(strfmt,
obj, ...)` will no longer support to use 0$ to specify the first argument, the first argument
should always reference by 1$ when use argument index to indicating the position of the argument
in the argument list.


Since Spark 3.3, the unbase64 function throws error for a malformed str input. Use
try_to_binary(<str>, 'base64') to tolerate malformed input and return NULL instead. In Spark 3.2
and earlier, the unbase64 function returns a best-efforts result for a malformed str input.
```

Basierend auf der Zusammenfassung werden von AWS Glue vier Änderungen vorgeschlagen, um das Skript erfolgreich von AWS Glue 2.0 auf Glue 4.0 zu AWS aktualisieren:

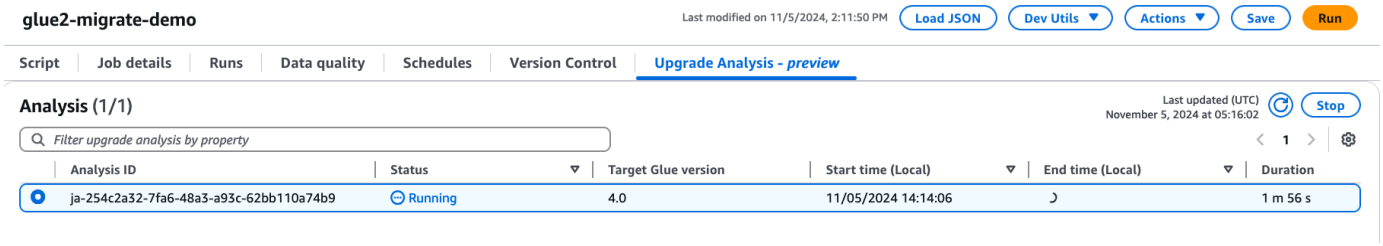
1. Spark-SQL-Konfiguration (spark.sql.adaptive.enabled): Diese Änderung dient der Wiederherstellung des Anwendungsverhaltens, da ab Spark 3.2 eine neue Funktion für die Ausführung adaptiver Abfragen in Spark SQL eingeführt wird. Sie können diese Konfigurationsänderung überprüfen und sie je nach Wunsch weiter aktivieren oder deaktivieren.
2. DataFrame API-Änderung: Die Pfadoption kann nicht zusammen mit anderen DataFrameReader Vorgängen wie `load()` existieren. Um das vorherige Verhalten beizubehalten, hat AWS Glue das Skript aktualisiert und eine neue SQL-Konfiguration hinzugefügt (spark.sql.legacy). pathOptionBehavior.aktiviert).

3. Änderung der Spark-SQL-API: Das Verhalten von `strfmt` in `format_string(strfmt, obj, ...)` wurde aktualisiert und ist nun nicht mehr `0$` das erste Argument. Um die Kompatibilität zu gewährleisten, hat AWS Glue das Skript so geändert, dass es stattdessen `1$` als erstes Argument verwendet wird.
4. Änderung der Spark-SQL-API: Die `unbase64` Funktion erlaubt keine fehlerhaften Zeichenketteneingaben. Um das vorherige Verhalten beizubehalten, hat AWS Glue das Skript aktualisiert, um die `try_to_binary` Funktion zu verwenden.

Eine laufende Upgrade-Analyse wird gestoppt

Sie können eine laufende Upgrade-Analyse abbrechen oder die Analyse einfach beenden.

1. Wählen Sie die Registerkarte „Analyse aktualisieren“.
2. Wählen Sie den Job aus, der gerade ausgeführt wird, und klicken Sie dann auf Beenden. Dadurch wird die Analyse beendet. Anschließend können Sie eine weitere Upgrade-Analyse für denselben Job ausführen.



Überlegungen

Wenn Sie in der Vorschauphase mit der Nutzung von Spark-Upgrades beginnen, müssen Sie mehrere wichtige Aspekte berücksichtigen, um den Service optimal nutzen zu können.

- Serviceumfang und Einschränkungen: Die Vorschauversion konzentriert sich auf PySpark Code-Upgrades von AWS Glue-Versionen 2.0 auf Version 5.0. Derzeit verarbeitet der Service PySpark Code, der nicht auf zusätzlichen Bibliotheksabhängigkeiten angewiesen ist. Sie können automatisierte Upgrades für bis zu 10 Jobs gleichzeitig in einem AWS Konto ausführen, sodass Sie mehrere Jobs effizient aktualisieren und gleichzeitig die Systemstabilität aufrechterhalten können.
  - Es werden nur PySpark Jobs unterstützt.
  - Die Upgrade-Analyse läuft nach 24 Stunden ab.

  - Für einen Job kann jeweils nur eine aktive Upgrade-Analyse ausgeführt werden. Auf Kontoebene können bis zu 10 aktive Upgrade-Analysen gleichzeitig ausgeführt werden.
- Optimierung der Kosten während des Upgrade-Prozesses: Da Spark-Upgrades generative KI verwenden, um den Upgrade-Plan über mehrere Iterationen hinweg zu validieren, wobei jede Iteration als AWS Glue-Job in Ihrem Konto ausgeführt wird, ist es wichtig, die Konfigurationen der Validierungsjob-Ausführung aus Kostengründen zu optimieren. Um dies zu erreichen, empfehlen wir, beim Start einer Upgrade-Analyse eine Run-Konfiguration wie folgt anzugeben:
  - Verwenden Sie Entwicklerkonten, die nicht zur Produktion gehören, und wählen Sie für die Validierung mit Spark-Upgrades Beispieldatensätze aus, die Ihre Produktionsdaten repräsentieren, aber kleiner sind.
  - Verwenden Sie Rechenressourcen mit der richtigen Größe, wie z. B. G.1X-Worker, und wählen Sie eine angemessene Anzahl von Workern für die Verarbeitung Ihrer Beispieldaten aus.
  - Aktivieren AWS Sie gegebenenfalls die auto-scaling von Glue-Jobs, um Ressourcen automatisch an die Arbeitslast anzupassen.

  Wenn Ihr Produktionsjob beispielsweise Terabyte an Daten mit 20 G.2X-Workern verarbeitet, können Sie den Upgrade-Job so konfigurieren, dass er einige Gigabyte repräsentativer Daten mit 2 G.2X-Workern verarbeitet und auto-scaling für die Validierung aktiviert ist.
- Bewährte Methoden in der Vorschau: Während der Vorschauphase empfehlen wir dringend, Ihr Upgrade mit Aufträgen zu beginnen, die nicht zur Produktion gehören. Dieser Ansatz ermöglicht es Ihnen, sich mit dem Upgrade-Workflow vertraut zu machen und zu verstehen, wie der Service mit verschiedenen Arten von Spark-Codemustern umgeht.
- Alarme und Benachrichtigungen: Wenn Sie die Generative AI-Upgrade-Funktion für einen Job verwenden, stellen Sie sicher, dass die Jobläufe alarms/notifications für fehlgeschlagene Jobs ausgeschaltet sind. Während des Upgrade-Vorgangs kann es in Ihrem Konto bis zu 10 fehlgeschlagene Jobläufe geben, bevor die aktualisierten Artefakte bereitgestellt werden.
- Regeln zur Erkennung von Anomalien: Deaktivieren Sie alle Regeln zur Erkennung von Anomalien für den Job, der ebenfalls aktualisiert wird, da die Daten, die während zwischengeschalteter Jobausführungen in die Ausgabeordner geschrieben werden, möglicherweise nicht das erwartete Format haben, während die Upgrade-Validierung läuft.
- Verwenden Sie die Upgrade-Analyse für idempotente Jobs: Verwenden Sie die Upgrade-Analyse für idempotente Jobs, um sicherzustellen, dass jeder nachfolgende Ausführungsversuch eines Validierungsauftrags dem vorherigen ähnelt und keine Probleme auftreten. Bei idempotenten Jobs handelt es sich um Jobs, die mehrfach mit denselben Eingabedaten ausgeführt werden können und jedes Mal dieselbe Ausgabe erzeugen. Wenn Sie die Generative AI-Upgrades für

Apache Spark in AWS Glue verwenden, führt der Service im Rahmen des Validierungsprozesses mehrere Iterationen Ihres Jobs aus. Bei jeder Iteration werden Änderungen an Ihrem Spark-Code und Ihren Konfigurationen vorgenommen, um den Upgrade-Plan zu validieren. Wenn Ihr Spark-Job nicht idempotent ist, kann es zu Problemen kommen, wenn Sie ihn mehrmals mit denselben Eingabedaten ausführen.

Regionsübergreifende Inferenz bei Spark-Upgrades

Spark Upgrades basiert auf regionsübergreifender Inferenz (CRIS) Amazon Bedrock und nutzt diese. Mit CRIS wählt Spark Upgrades automatisch die optimale Region innerhalb Ihrer Region aus (wie hier ausführlicher beschrieben), um Ihre Inferenzanfrage zu bearbeiten, die verfügbaren Rechenressourcen und die Modellverfügbarkeit zu maximieren und das beste Kundenerlebnis zu bieten. Für die Nutzung regionsübergreifender Inferenz fallen keine zusätzlichen Kosten an.

Regionsübergreifende Inferenzanfragen werden in den AWS Regionen gespeichert, die Teil der Region sind, in der sich die Daten ursprünglich befinden. Beispielsweise wird eine in den USA gestellte Anfrage innerhalb der AWS Regionen in den USA aufbewahrt. Obwohl die Daten nur in der Primärregion gespeichert bleiben, können sich Ihre Eingabeaufforderungen und Ausgabeergebnisse bei Verwendung regionsübergreifender Inferenz möglicherweise außerhalb Ihrer primären Region bewegen. Alle Daten werden verschlüsselt über das sichere Netzwerk von Amazon übertragen.

Arbeiten mit Spark-Jobs in AWS Glue

Enthält Informationen AWS Glue zu Spark-ETL-Jobs.

Themen

- Verwenden von Auftragsparametern in AWS Glue-Jobs
- AWS Glue Spark und PySpark Jobs
- AWS Glue Typen von Arbeitern
- ETL-Jobs streamen AWS Glue
- Datensatzabgleich mit AWS Lake Formation FindMatches
- Migrieren Sie Apache Spark-Programme zu AWS Glue

Verwenden von Auftragsparametern in AWS Glue-Jobs

Beim Erstellen eines AWS Glue-Jobs legen Sie einige Standardfelder fest, z. B. `Role` und`WorkerType`. Über die `Argument`-Felder (Auftragsparameter in der Konsole) können Sie zusätzliche Konfigurationsinformationen bereitstellen. In diesen Feldern können Sie AWS Glue-Jobs mit den in diesem Thema aufgeführten Argumenten (Parametern) versehen.

Weitere Informationen zur AWS Glue Job API finden Sie unterthe section called “Aufträge”.

> **Note**
>
> Job-Argumente haben eine maximale Größenbeschränkung von 260 KB. Eine Gültigkeitsprüfung führt zu einem Fehler, wenn die Argumentgröße größer als 260 KB ist.

Festlegen der Auftragsparameter

Sie können einen Auftrag über die Konsole auf der Registerkarte Job details (Details zur Auftragsausführung) unter der Kopfzeile Job Parameters (Auftragsparameter) konfigurieren. Sie können einen Job auch über die Einstellung AWS CLI by `DefaultArguments` oder `NonOverridableArguments` on a job oder setting `Arguments` on a job run konfigurieren. Für den Auftrag festgelegte Argumente werden bei jeder Ausführung des Auftrags übergeben, während bei der Auftragsausführung festgelegte Argumente nur für diese einzelne Ausführung übergeben werden.

Im Folgenden finden Sie beispielsweise die Syntax zum Ausführen eines Auftrags mit `--arguments` zum Festlegen eines Auftragsparameters.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py"'
```

Zugreifen auf Auftragsparameter

Beim Schreiben von AWS Glue-Skripten möchten Sie möglicherweise auf Job-Parameterwerte zugreifen, um das Verhalten Ihres eigenen Codes zu ändern. In unseren Bibliotheken stellen wir hierfür Hilfsmethoden zur Verfügung. Diese Methoden lösen Parameterwerte der Auftragsausführung auf, die die Parameterwerte des Auftrags überschreiben. Bei der Auflösung von Parametern, die an mehreren Stellen festgelegt wurden, überschreibt Auftrag `NonOverridableArguments` die Auftragsausführung `Arguments`, die wiederum Auftrag `DefaultArguments` überschreibt.

In Python:

In Python-Aufträgen stellen wir eine Funktion mit dem Namen `getResolvedParameters` bereit. Weitere Informationen finden Sie unter the section called “getResolvedOptions”. Die Auftragsparameter sind in der Variablen `sys.argv` verfügbar.

In Scala:

In Scala-Aufträgen stellen wir ein Objekt mit dem Namen `GlueArgParser` bereit. Weitere Informationen finden Sie unter the section called “GlueArgParser”. Die Auftragsparameter sind in der Variablen `sysArgs` verfügbar.

Referenz der Auftragsparameter

AWS Glue erkennt die folgenden Argumentnamen, die Sie verwenden können, um die Skriptumgebung für Ihre Jobs und Jobausführungen einzurichten:

`--additional-python-modules`

Eine durch Kommas getrennte Liste, die eine Reihe von Python-Paketen darstellt, die installiert werden sollen. Sie können Pakete von PyPI installieren oder eine benutzerdefinierte Verteilung bereitstellen. Ein PyPI-Paketeintrag liegt im Format *`package`*`==`*`version`* vor und enthält den PyPI-Namen und die Version Ihres Zielpakets. Ein benutzerdefinierter Verteilungseintrag ist der S3-Pfad zur Verteilung.

Einträge verwenden einen Python-Versionsabgleich, um Paket und Version abzugleichen. Dies bedeutet, dass Sie zwei Gleichheitszeichen verwenden müssen, z. B. ==. Es gibt andere Operatoren für den Versionsabgleich. Weitere Informationen dazu finden Sie unter PEP 440.

Um Optionen für die Modulinstallation an `pip3` zu übergeben, verwenden Sie den --python-modules-installer-option-Parameter.

`--auto-scale-within-microbatch`

Der Standardwert ist "True". Dieser Parameter kann nur für AWS Glue-Streaming-Jobs verwendet werden, bei denen die Streaming-Daten in einer Reihe von Mikrobatches verarbeitet werden. Auto Scaling muss aktiviert sein. Wenn dieser Wert auf falsch gesetzt ist, wird der exponentielle gleitende Durchschnitt der Batch-Dauer für abgeschlossene Micro-Batches berechnet. Dieser Wert wird mit der Fenstergröße verglichen, um festzulegen, ob die Anzahl der Ausführer hoch- oder herunterskaliert werden soll. Die Skalierung erfolgt erst, wenn ein Mikro-Batch

abgeschlossen ist. Wenn dieser Wert während eines Mikro-Batches auf wahr gesetzt ist, wird er hochskaliert, wenn die Anzahl der Spark-Aufgaben 30 Sekunden lang gleich bleibt oder die aktuelle Batch-Verarbeitung größer als die Fenstergröße ist. Die Anzahl der Ausführer sinkt, wenn ein Ausführer länger als 60 Sekunden im Leerlauf war oder der exponentielle gleitende Durchschnitt der Batch-Dauer niedrig ist.

`--class`

Die Scala-Klasse, die als Einstiegspunkt für Ihr Scala-Skript dient. Dies gilt nur, wenn die `--job-language` auf `scala` eingestellt ist.

`--continuous-log-conversionPattern`

Gibt ein benutzerdefiniertes Konvertierungsprotokollmuster für einen Auftrag an, der für die kontinuierliche Protokollierung aktiviert ist. Das Konvertierungsmuster gilt nur für Treiberprotokolle und Executor-Protokolle. Es hat keinen Einfluss auf den AWS Glue-Fortschrittsbalken.

`--continuous-log-logGroup`

Gibt einen benutzerdefinierten CloudWatch Amazon-Protokollgruppennamen für einen Job an, der für die kontinuierliche Protokollierung aktiviert ist.

`--continuous-log-logStreamPrefix`

Gibt ein benutzerdefiniertes CloudWatch Protokollstream-Präfix für einen Job an, der für die kontinuierliche Protokollierung aktiviert ist.

`--customer-driver-env-vars` und **`--customer-executor-env-vars`**

Mit diesen Parametern werden Umgebungsvariablen auf dem Betriebssystem für jeden Worker (Treiber oder Executor) festgelegt. Sie können diese Parameter verwenden, wenn Sie Plattformen und benutzerdefinierte Frameworks auf AWS Glue aufbauen, damit Ihre Benutzer Jobs darauf schreiben können. Wenn Sie diese beiden Flags aktivieren, können Sie unterschiedliche Umgebungsvariablen für den Treiber bzw. den Executor festlegen, ohne dieselbe Logik in das Job-Skript selbst einfügen zu müssen.

Beispielverwendung

Im Folgenden finden Sie ein Beispiel für die Verwendung dieser Parameter:

```
"–customer-driver-env-vars", "CUSTOMER_KEY1=VAL1,CUSTOMER_KEY2=\"val2,val2 val2\"",
"–customer-executor-env-vars", "CUSTOMER_KEY3=VAL3,KEY4=VAL4"
```

Das Setzen dieser Werte im Jobrun-Argument entspricht der Ausführung der folgenden Befehle:

Im Treiber:

- exportiere CUSTOMER_ = KEY1 VAL1
- exportiere CUSTOMER_ KEY2 ="val2, val2 val2"

Im Executor:

- exportiere CUSTOMER_ = KEY3 VAL3

Anschließend können Sie im Jobskript selbst die Umgebungsvariablen mit `os.environ.get("CUSTOMER_KEY1")` oder abrufen. `System.getenv("CUSTOMER_KEY1")`

Erzwungene Syntax

Beachten Sie bei der Definition von Umgebungsvariablen die folgenden Standards:

- Jeder Schlüssel muss den haben`CUSTOMER_ prefix`.

  Zum Beispiel: for`"CUSTOMER_KEY3=VAL3,KEY4=VAL4"`, `KEY4=VAL4` wird ignoriert und nicht gesetzt.
- Jedes Schlüssel- und Wertepaar muss durch ein einzelnes Komma abgegrenzt werden.

  Beispiel: `"CUSTOMER_KEY3=VAL3,CUSTOMER_KEY4=VAL4"`
- Wenn der „Wert“ Leerzeichen oder Kommas enthält, muss er in Anführungszeichen definiert werden.

  Beispiel: `CUSTOMER_KEY2=\"val2,val2 val2\"`

Diese Syntax ist den Standards für das Setzen von Bash-Umgebungsvariablen sehr ähnlich.

`--datalake-formats`

Wird in AWS Glue 3.0 und späteren Versionen unterstützt.

Gibt das zu verwendende Data Lake-Framework an. AWS Glue fügt die erforderlichen JAR-Dateien für die von Ihnen angegebenen Frameworks in die ein`classpath`. Weitere Informationen finden Sie unter Verwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

Sie können einen oder mehrere der folgenden Werte durch Komma getrennt angeben:

- `hudi`

- `delta`
- `iceberg`

Übergeben Sie beispielsweise das folgende Argument, um alle drei Frameworks anzugeben.

```
'--datalake-formats': 'hudi,delta,iceberg'
```

`--disable-proxy-v2`

Deaktivieren Sie den Service-Proxy, um AWS Serviceanrufe an Amazon S3 und AWS Glue CloudWatch, die von Ihrem Skript ausgehen, über Ihre VPC zuzulassen. Weitere Informationen finden Sie unter Konfigurieren von AWS -Aufrufen, um Ihre VPC durchlaufen . Um den Service-Proxy zu deaktivieren, legen Sie den Wert dieses Parameters auf `true` fest.

`--enable-auto-scaling`

Aktiviert Auto Scaling und die Abrechnung pro Worker, wenn der Wert auf `true` eingestellt wird.

`--enable-continuous-cloudwatch-log`

Ermöglicht die kontinuierliche Protokollierung von AWS Glue-Jobs in Echtzeit. Sie können Apache Spark-Aufgabenprotokolle in Echtzeit in CloudWatch anzeigen.

`--enable-continuous-log-filter`

Gibt einen Standardfilter (`true`) oder keinen Filter (`false`) an, wenn Sie eine Aufgabe erstellen oder bearbeiten, die für die kontinuierliche Protokollierung aktiviert ist. Wenn Sie den Standardfilter wählen, werden unbrauchbare Apache Spark driver/executor - und Apache Hadoop YARN-Heartbeat-Protokollnachrichten gelöscht. Wenn Sie keinen Filter auswählen, erhalten Sie alle Protokollmeldungen.

`--enable-glue-datacatalog`

Ermöglicht es Ihnen, den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore zu verwenden. Um dieses Feature zu aktivieren, setzen Sie den Wert auf `true`.

`--enable-job-insights`

Ermöglicht zusätzliche Fehleranalyseüberwachung mit AWS Glue Job Run Insights. Details hierzu finden Sie unter the section called “Überwachung mit Erkenntnissen zur AWS Glue Auftragsausführung”. Der Wert ist standardmäßig auf `true` festgelegt und Auftragsausführungs-Insights sind aktiviert.

Diese Option ist für AWS Glue Version 2.0 und 3.0 verfügbar.

`--enable-lakeformation-fine-grained-access`

Ermöglicht eine feinkörnige Zugriffskontrolle für AWS Klebeaufträge. Weitere Informationen finden Sie unter the section called “Lake Formation für FGAC”.

`--enable-metrics`

Aktiviert die Erfassung von Metriken für die Erstellung von Auftragsprofilen für diese Auftragsausführung. Diese Metriken sind auf der AWS Glue-Konsole und der CloudWatch Amazon-Konsole verfügbar. Der Wert dieses Parameters ist nicht relevant. Um dieses Feature zu aktivieren, können Sie für diesen Parameter einen beliebigen Wert angeben. Aus Gründen der Übersichtlichkeit wird jedoch `true` empfohlen. Um dieses Feature zu deaktivieren, entfernen Sie diesen Parameter aus Ihrer Auftragskonfiguration.

`--enable-observability-metrics`

Ermöglicht eine Reihe von Observability-Metriken, um Erkenntnisse darüber zu gewinnen, was in jedem Job passiert, der auf der Seite Job Runs Monitoring unter der AWS Glue-Konsole und der Amazon CloudWatch Konsole ausgeführt wird. Um dieses Feature zu aktivieren, setzen Sie den Wert dieses Parameters auf „true“. Um dieses Feature zu deaktivieren, setzen Sie ihn auf `false` oder entfernen Sie den Parameter aus der Auftragskonfiguration.

`--enable-rename-algorithm-v2`

Setzt die Version des EMRFS-Umbenennungsalgorithmus auf Version 2. Wenn ein Spark-Auftrag den dynamischen Partitionsüberschreibungsmodus verwendet, besteht die Möglichkeit, dass eine doppelte Partition erstellt wird. Zum Beispiel können Sie eine doppelte Partition wie `s3://bucket/table/location/p1=1/p1=1` erhalten. Hier ist P1 die Partition, die überschrieben wird. Das Umbenennen des Algorithmus Version 2 behebt dieses Problem.

Diese Option ist nur in der AWS Glue-Version 1.0 verfügbar.

`--enable-s3-parquet-optimized-committer`

Aktiviert den EMRFS-S3-optimierten Committer zum Schreiben von Parquet-Daten in Amazon S3. Sie können das parameter/value Paar über die AWS Glue-Konsole angeben, wenn Sie einen AWS Glue-Job erstellen oder aktualisieren. Durch Festlegen des Werts **`true`** wird der Committer aktiviert. Standardmäßig ist die Flagge in AWS Glue 3.0 aktiviert und in AWS Glue 2.0 ausgeschaltet.

Weitere Informationen finden Sie unter Verwenden der EMRFS S3-optimierten Committer-Klasse.

--enable-spark-ui

Wenn diese Option auf gesetzt ist`true`, wird die Funktion aktiviert, die Spark-Benutzeroberfläche zum Überwachen und Debuggen von AWS Glue-ETL-Jobs zu verwenden.

--executor-cores

Anzahl der Spark-Aufgaben, die parallel ausgeführt werden können. Diese Option wird auf AWS Glue 3.0+ unterstützt. Der Wert sollte das Zweifache der Zahl von v CPUs für den Worker-Typ nicht überschreiten, also 8 an`G.1X`, 16 an`G.2X`, 32 an`G.4X`, 64 an`G.8X`, 96 an`G.12X`, 128 an `G.16X` und 8 an`R.1X`, 16 on`R.2X`, 32 on`R.4X`, 64 on`R.8X`. Sie sollten bei der Aktualisierung dieser Konfiguration vorsichtig sein, da sie sich auf die Leistung des Auftrags auswirken könnte. Eine erhöhte Parallelität der Aufgaben führt nämlich zu Speicher- und Festplattendruck sowie zu einer Drosselung der Quell- und Zielsysteme (z. B. würde dies zu mehr gleichzeitigen Verbindungen in Amazon RDS führen).

--extra-files

Die Amazon S3 S3-Pfade zu zusätzlichen Dateien, wie Konfigurationsdateien, die AWS Glue vor der Ausführung in das Arbeitsverzeichnis Ihres Skripts auf dem Treiberknoten kopiert. Mehrere Werte müssen vollständige Pfade sein, die durch Kommas (`,`) getrennt werden. Es werden nur einzelne Dateien unterstützt, kein Verzeichnispfad. Diese Option wird für Python-Shell-Auftragstypen nicht unterstützt.

--extra-jars

Die Amazon S3 S3-Pfade zu zusätzlichen Dateien, die AWS Glue auf den Treiber und die Executoren kopiert. AWS Glue fügt diese Dateien auch dem Java-Klassenpfad hinzu, bevor Ihr Skript ausgeführt wird. Mehrere Werte müssen vollständige Pfade sein, die durch Kommas (`,`) getrennt werden. Die Erweiterung muss nicht sein `.jar`

--extra-py-files

Die Amazon S3 S3-Pfade zu zusätzlichen Python-Modulen, die AWS Glue dem Python-Pfad auf dem Treiberknoten hinzufügt, bevor Ihr Skript ausgeführt wird. Mehrere Werte müssen vollständige Pfade sein, die durch Kommas (`,`) getrennt werden. Es werden nur einzelne Dateien unterstützt, kein Verzeichnispfad.

--job-bookmark-option

Steuert die Darstellung eines Auftrags-Lesezeichens. Die folgenden Optionswerte können festgelegt werden:

--job-bookmark-option-Wert	Beschreibung
`job-bookmark-enable`	Bereits verarbeitete Daten nachverfolgen. Wenn ein Auftrag ausgeführt wird, werden neue Daten seit dem letzten Checkpoint verarbeitet.
`job-bookmark-disable`	Immer das gesamte Dataset verarbeiten. Sie sind für die Verwaltung der Ausgaben aus früheren Auftragsa usführungen verantwortlich.
`job-bookmark-pause`	Verarbeiten inkrementeller Daten seit der letzten erfolgreichen Ausführung oder der Daten in dem durch die folgenden Unteroptionen identifizierten Bereich, ohne den Status des letzten Lesezeichens zu aktualisi eren. Sie sind für die Verwaltung der Ausgaben aus früheren Auftragsausführungen verantwortlich. Die beiden Unteroptionen lauten wie folgt: • **`job-bookmark-from`** `<from-value>` ist die Ausführungs-ID, die alle Eingaben darstellt, die bis zur letzten erfolgreichen Ausführung vor und einschließlich der angegebenen Ausführungs-ID verarbeitet wurden. Die entsprechende Eingabe wird ignoriert. • **`job-bookmark-to`** `<to-value>` ist die Ausführungs-ID, die alle Eingaben darstellt, die bis zur letzten erfolgreichen Ausführung vor und einschließlich der angegebenen Ausführungs-ID verarbeitet wurden. Die entsprechende Eingabe ohne die Eingabe, die durch den `<from-value>` identifiz iert wird, wird von der Aufgabe verarbeitet. Jede Eingabe, die später als diese Eingabe ist, wird auch aus der Verarbeitung ausgeschlossen. Der Auftrags-Lesezeichenstatus wird nicht aktualisiert, wenn dieser Optionssatz angegeben wird.

--job-bookmark-option-Wert	Beschreibung
	Die Unteroptionen sind optional. Bei ihrer Verwendung müssen jedoch beide Unteroptionen angegeben werden.

Um beispielsweise ein Aufgabenlesezeichen zu aktivieren, übergeben Sie das folgende Argument:

```
'--job-bookmark-option': 'job-bookmark-enable'
```

`--job-language`

Die Skript-Programmiersprache. Dieser Wert muss entweder `scala` oder `python` sein. Wenn dieser Parameter nicht vorhanden ist, ist der Standardwert `python`.

`--python-modules-installer-option`

Eine Klartextzeichenfolge, die Optionen definiert, die an `pip3` übergeben werden, wenn Module mit --additional-python-modules installiert werden. Stellen Sie Optionen wie in der Befehlszeile bereit, getrennt durch Leerzeichen und mit vorangestellten Bindestrichen. Weitere Informationen zur Verwendung finden Sie unter the section called “Installation zusätzlicher Python-Module mit Pip in AWS Glue 2.0 oder höher”.

> **Note**
>
> Diese Option wird für AWS Glue-Jobs nicht unterstützt, wenn Sie Python 3.9 verwenden.

`--scriptLocation`

Der Amazon Simple Storage Service (Amazon S3)-Speicherort, an dem Ihr ETL-Skript abgelegt ist (im Format `s3://path/to/my/script.py`). Dieser Parameter überschreibt einen Skript-Speicherort im `JobCommand`-Objekt.

`--spark-event-logs-path`

Gibt einen Amazon-S3-Pfad an. Wenn Sie die Spark-UI-Überwachungsfunktion verwenden, bereinigt AWS Glue die Spark-Ereignisprotokolle alle 30 Sekunden in diesem Amazon-S3-Pfad

über einen Bucket, der als temporäres Verzeichnis zum Speichern von Spark-UI-Ereignissen verwendet werden kann.

`--TempDir`

Gibt einen Amazon-S3-Pfad zu einem Bucket an, der als temporäres Verzeichnis für den Auftrag verwendet werden kann.

Um beispielsweise ein temporäres Verzeichnis zu setzen, übergeben Sie das Argument:

```
'--TempDir': 's3-path-to-directory'
```

> **Note**
>
> AWS Glue erstellt einen temporären Bucket für Jobs, falls noch kein Bucket in einer Region existiert. Dieser Bucket erlaubt möglicherweise den öffentlichen Zugriff. Sie können entweder den Bucket in Amazon S3 ändern, um die öffentliche Zugriffssperre festzulegen, oder den Bucket später löschen, nachdem alle Aufträge in dieser Region abgeschlossen sind.

`--use-postgres-driver`

Wenn Sie diesen Wert auf `true` setzen, wird der Postgres-JDBC-Treiber im Klassenpfad priorisiert, um einen Konflikt mit dem Amazon-Redshift-JDBC-Treiber zu vermeiden. Diese Option ist nur in AWS Glue Version 2.0 verfügbar.

`--user-jars-first`

Wenn Sie diesen Wert auf `true` setzen, werden die zusätzlichen JAR-Dateien des Kunden im Klassenpfad priorisiert. Diese Option ist nur in AWS Glue Version 2.0 oder höher verfügbar.

`--conf`

Sie steuert die Spark-Konfigurationsparameter. Sie ist für fortschrittliche Anwendungsfälle.

`--encryption-type`

Legacy-Parameter. Das entsprechende Verhalten sollte über Sicherheitskonfigurationen konfiguriert werden. Weitere Informationen zu Sicherheitskonfigurationen finden Sie unter the section called “Verschlüsselung von Daten, die geschrieben wurden von AWS Glue”.

AWS Glue verwendet intern die folgenden Argumente und du solltest sie niemals verwenden:

- `--debug`— AWS Innenverklebungen. Nicht einstellen.
- `--mode`— AWS Innenverklebungen. Nicht einstellen.
- `--JOB_NAME`— AWS Innenverklebungen. Nicht einstellen.
- `--endpoint`— AWS Innenverklebungen. Nicht einstellen.

AWS Glue unterstützt das Bootstrapping einer Umgebung mit dem `site` Python-Modul, mit dem standortspezifische `sitecustomize` Anpassungen vorgenommen werden können. Das Bootstrapping Ihrer eigenen Initialisierungsfunktionen wird nur für fortgeschrittene Anwendungsfälle empfohlen und wird auf Glue 4.0 nach bestem Wissen unterstützt. AWS

Das Präfix der Umgebungsvariablen, `GLUE_CUSTOMER`, ist für die Verwendung durch Kunden reserviert.

AWS Glue Spark und PySpark Jobs

AWS Glue unterstützen Sie Spark und PySpark Jobs. Ein Spark-Job wird in einer Apache Spark-Umgebung ausgeführt, die von verwaltet wird AWS Glue. Er verarbeitet Daten in Batches. Ein Streaming-ETL-Auftrag ähnelt einem Spark-Auftrag, mit der Ausnahme, dass er ETL für Datenstreams ausführt. Er verwendet das Framework Apache Spark Structured Streaming. Einige Features von Spark-Aufträgen sind für Streaming-ETL-Aufträge nicht verfügbar.

Die folgenden Abschnitte enthalten Informationen zu AWS Glue Spark und PySpark Jobs.

Themen

- Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue
- Spark-Skripte in der AWS Glue-Konsole bearbeiten
- Aufgräge (veraltet)
- Verfolgen von verarbeiteten Daten mit Auftragslesezeichen
- Speichern von Spark-Shuffle-Daten
- Überwachen AWS Glue Stellen bei Spark
- Generative KI-Fehlerbehebung für Apache Spark in AWS Glue

Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue

Wenn Sie Ihren Job auf der AWS Glue Konsole definieren, geben Sie Werte für Eigenschaften zur Steuerung der AWS Glue Laufzeitumgebung an.

Definieren von Auftragseigenschaften für Spark-Aufträge

Die folgende Liste enthält Beschreibungen der Eigenschaften eines Spark-Auftrags. Informationen zu den Eigenschaften von Python-Shell-Aufträgen finden Sie unter Definieren von Auftragseigenschaften für Python-Shell-Aufträge. Informationen zu den Eigenschaften eines Streaming-ETL-Auftrags finden Sie unter the section called “Definieren von Auftragseigenschaften für einen Streaming-ETL-Auftrag”.

Die Eigenschaften werden in der Reihenfolge aufgeführt, in der sie im Assistenten zum Hinzufügen von Jobs auf der AWS Glue Konsole angezeigt werden.

Name

Geben Sie eine UTF-8-Zeichenfolge mit einer maximalen Länge von 255 Zeichen an.

Beschreibung

Geben Sie optional eine Beschreibung mit bis zu 2048 Zeichen ein.

IAM Role (IAM-Rolle)

Geben Sie die IAM-Rolle an, die für die Autorisierung von Ressourcen verwendet wird, die für die Ausführung des Auftrags und den Zugriff auf Datenspeicher verwendet werden. Weitere Informationen zu den Berechtigungen für die Ausführung von Jobs in finden Sie AWS Glue unterIdentitäts- und Zugriffsmanagement für AWS Glue.

Typ

Der Typ des ETL-Jobs. Dies wird automatisch basierend auf dem Typ der ausgewählten Datenquellen festgelegt.

- Spark führt ein Apache Spark-ETL-Skript mit dem Job-Befehl aus`glueetl`.
- Spark Streaming führt ein Apache Spark-Streaming-ETL-Skript mit dem Job-Befehl aus`gluestreaming`. Weitere Informationen finden Sie unter the section called “Streaming-ETL-Aufträge”.
- Die Python-Shell führt ein Python-Skript mit dem Job-Befehl aus`pythonshell`. Weitere Informationen finden Sie unter Konfiguration von Jobeigenschaften für Python-Shell-Jobs in AWS Glue.

AWS Glue -Version

AWS Glue Version bestimmt die Versionen von Apache Spark und Python, die für den Job verfügbar sind, wie in der folgenden Tabelle angegeben.

AWS Glue Version	Unterstützte Spark- und Python-Versionen
5.0	• Spark 3.5.4 • Python 3.11
4,0	• Spark 3.3.0 • Python 3.10
3.0	• Spark 3.1.1 • Python 3.7

Sprache

Der Code im ETL-Skript definiert die Logik Ihres Auftrags. Das Skript kann in Python oder Scala programmiert werden. Sie können wählen, ob das Skript, das der Job ausführt, von Ihnen generiert AWS Glue oder bereitgestellt wird. Geben Sie den Skript-Namen und den Speicherort in Amazon Simple Storage Service (Amazon S3) an. Vergewissern Sie sich, dass sich keine Datei mit dem Namen des Skriptverzeichnisses im Pfad befindet. Weitere Informationen zum Schreiben von Skripts finden Sie unter AWS Glue Leitfaden zur Programmierung.

Worker type (Worker-Typ)

Die folgenden Worker-Typen sind verfügbar:

Die Ressourcen, die den AWS Glue Mitarbeitern zur Verfügung stehen, werden in gemessen DPUs. Eine DPU ist ein relatives Maß für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt.

- **`G.025X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker entspricht 0,25 DPU (2 VCPUs, 4 GB Arbeitsspeicher) mit 84 GB Festplatte (ca. 34 GB freier Speicher). Wir empfehlen diesen Worker-Typ für Streaming-Aufträge mit geringem Volumen. Dieser Worker-Typ ist nur für Streaming-Jobs der AWS Glue Version 3.0 oder höher verfügbar.

- **`G.1X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 94 GB Festplatte (ca. 44 GB freier Speicher) zugeordnet. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
- **`G.2X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 138 GB Festplatte (ca. 78 GB freier Speicherplatz) zu. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
- **`G.4X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte (ca. 230 GB freier Speicher) zu. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten.
- **`G.8X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte (ca. 485 GB freier Speicher) zu. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten.
- **`G.12X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ordnet 12 DPU (48 VCPUs, 192 GB Arbeitsspeicher) mit 768 GB Festplatte (ca. 741 GB freier Speicher) zu. Wir empfehlen diesen Workertyp für Jobs mit sehr großen und ressourcenintensiven Workloads, die eine erhebliche Rechenkapazität erfordern.
- **`G.16X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker weist 16 DPU (64 VCPUs, 256 GB Arbeitsspeicher) mit 1024 GB Festplatte (ca. 996 GB frei) zu. Wir empfehlen diesen Workertyp für Jobs mit den größten und ressourcenintensivsten Workloads, die maximale Rechenkapazität erfordern.
- **`R.1X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist einer DPU mit speicheroptimierter Konfiguration zugeordnet. Wir empfehlen diesen Workertyp für speicherintensive Workloads, bei denen häufig Fehler auftreten out-of-memory oder bei denen hohe Raten erforderlich sind. memory-to-CPU
- **`R.2X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist 2 DPU mit speicheroptimierter Konfiguration zugeordnet. Wir

empfehlen diesen Workertyp für speicherintensive Workloads, bei denen häufig Fehler auftreten out-of-memory oder bei denen hohe Raten erforderlich sind. memory-to-CPU

- **`R.4X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist 4 DPU mit speicheroptimierter Konfiguration zugeordnet. Wir empfehlen diesen Workertyp für große speicherintensive Workloads, bei denen häufig Fehler auftreten out-of-memory oder bei denen hohe Raten erforderlich sind. memory-to-CPU
- **`R.8X`** – Wenn Sie diese Art auswählen, müssen Sie auch einen Wert für die Anzahl der Worker angeben. Jeder Worker ist 8 DPU mit speicheroptimierter Konfiguration zugeordnet. Wir empfehlen diesen Workertyp für sehr große speicherintensive Workloads, bei denen häufig Fehler auftreten out-of-memory oder hohe Raten erforderlich sind. memory-to-CPU

Spezifikationen für den Worker-Typ

Die folgende Tabelle enthält detaillierte Spezifikationen für alle verfügbaren G-Worker-Typen:

Spezifikationen des G-Worker-Typs

Art des Arbeitnehmers	DPU pro Knoten	vCPU	Speicher (GB)	Festplatte (GB)	Freier Festplattenspeicher (GB)	Spark-Executoren pro Knoten
G.025X	0,25	2	4	84	34	1
G.1X	1	4	16	94	44	1
G.2X	2	8	32	138	78	1
G.4X	4	16	64	256	230	1
G.8X	8	32	128	512	485	1
G12X	12	48	192	768	741	1
G.16X	16	64	256	1024	996	1

Wichtig: Die Workertypen G.12X und G.16X sowie alle R-Workertypen (R.1X bis R.8X) haben eine höhere Startlatenz.

Ihnen wird ein Stundensatz berechnet, der auf der Anzahl der für die Ausführung Ihrer ETL-Jobs verwendeten Jobs basiert. DPUs Weitere Informationen finden Sie in der AWS Glue Preisliste.

Wenn Sie für Jobs der AWS Glue Version 1.0 oder früher einen Job mithilfe der Konsole konfigurieren und den Worker-Typ Standard angeben, wird die maximale Kapazität festgelegt und die Anzahl der Worker wird zum Wert von Maximale Kapazität — 1. Wenn Sie das AWS Command Line Interface (AWS CLI) oder AWS SDK verwenden, können Sie den Parameter Max Capacity angeben, oder Sie können sowohl den Worker-Typ als auch die Anzahl der Worker angeben.

Für Jobs der AWS Glue Version 2.0 oder höher können Sie keine maximale Kapazität angeben. Stattdessen sollten Sie einen Worker-Typ und die Anzahl der Worker angeben.

`G.4X`und **`G.8X`** Arbeitertypen sind nur für Spark ETL-Jobs der AWS Glue Version 3.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Nordkalifornien), USA West (Oregon), Asien-Pazifik (Mumbai), Asien-Pazifik (Seoul), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland), Europa (London), Europa (Spanien), Europa (Stockholm) und Südamerika (São Paulo).

`G.12X` **`G.16X`**, und **`R.1X`** über **`R.8X`** Worker-Typen sind nur für Spark ETL-Jobs der AWS Glue Version 4.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Nord-Virginia), USA West (Oregon), USA Ost (Ohio), Europa (Irland) und Europa (Frankfurt). Weitere Regionen werden in future Versionen unterstützt.

Angeforderte Anzahl von Mitarbeitern

Für die meisten Worker-Typen müssen Sie die Anzahl der Worker angeben, die bei der Ausführung des Auftrags zugewiesen werden.

Auftrags-Lesezeichen

Geben Sie an, wie Statusinformationen AWS Glue verarbeitet werden, wenn der Job ausgeführt wird. Sie können bereits verarbeitete Daten beibehalten lassen, Statusinformationen aktualisieren oder Statusinformationen ignorieren. Weitere Informationen finden Sie unter the section called “Verfolgen von verarbeiteten Daten mit Auftragslesezeichen”.

Warteschlange beim Ausführen von Job

Gibt an, ob Auftragsausführungen in die Warteschlange gestellt werden, um sie später auszuführen, wenn sie aufgrund von Dienstkontingenten nicht sofort ausgeführt werden können.

Wenn diese Option aktiviert ist, ist die Warteschlange für die Auftragsausführungen aktiviert. Wenn sie nicht ausgefüllt sind, werden die Auftragsausführungen nicht in die Warteschlange aufgenommen.

Wenn diese Einstellung nicht mit dem Wert übereinstimmt, der in der Auftragsausführung festgelegt wurde, wird der Wert aus dem Feld für die Auftragsausführung verwendet.

Flexible Ausführung

Wenn Sie einen Job mit AWS Studio oder der API konfigurieren, können Sie eine Standard- oder flexible Jobausführungsklasse angeben. Ihre Aufträge können unterschiedliche Prioritäten und Zeitsensibilität haben. Die Standardausführungsklasse ist ideal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

Die flexible Ausführungsklasse eignet sich für nicht dringende Aufträge, deren Start- und Abschlusszeiten variieren können, z. B. für Vorproduktionsaufträge, Tests und einmalige Datenübertragungen. Flexible Auftragsausführungen werden für Jobs unterstützt, die AWS Glue Version 3.0 oder höher und/oder `G.1X` `G.2X` Workertypen verwenden. Die neuen Workertypen (`G.12X`G.16X, und `R.1X` bis`R.8X`) unterstützen keine flexible Ausführung.

Ausführungen von Flex-Aufträgen werden basierend auf der Anzahl der Worker abgerechnet, die zu einem beliebigen Zeitpunkt ausgeführt werden. Die Anzahl der Worker kann für die Ausführung eines flexiblen Auftrags hinzugefügt oder entfernt werden. Anstatt eine einfache Berechnung von `Max Capacity` * `Execution Time` abzurechnen, trägt jeder Worker für die Zeit bei, die er während der Ausführung des Auftrags ausgeführt hat. Die Rechnung ist die Summe von (`Number of DPUs per worker*time each worker ran`).

Weitere Informationen finden Sie im Hilfebereich in AWS Studio oder Aufträge undAuftragsausführungen.

Anzahl der Wiederholungen

Geben Sie an, wie oft (zwischen 0 und 10) der Job automatisch neu gestartet werden AWS Glue soll, falls er fehlschlägt. Aufträge, die das Timeout-Limit erreichen, werden nicht neu gestartet.

Zeitüberschreitung von Aufträgen

Legt die maximale Ausführungszeit in Minuten fest. Das Maximum ist 7 Tage oder 10.080 Minuten. Andernfalls lösen die Jobs eine Ausnahme aus.

Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

> **Bewährte Methoden für Job-Timeouts**
>
> Jobs werden auf der Grundlage der Ausführungszeit abgerechnet. Um unerwartete Gebühren zu vermeiden, konfigurieren Sie Timeout-Werte, die der erwarteten Ausführungszeit Ihres Jobs entsprechen.

Erweiterte Eigenschaften

Dateiname des Skripts

Ein eindeutiger Skriptname für Ihren Job. Kann nicht als Job ohne Titel bezeichnet werden.

Skriptpfad

Der Amazon S3 S3-Speicherort des Skripts. Der Pfad muss im Format `s3://bucket/prefix/path/` eingegeben werden. Es muss mit einem Schrägstrich (/) enden und darf keine Dateien enthalten.

Auftragsmetriken

Schalten Sie die Erstellung von CloudWatch Amazon-Metriken ein oder aus, wenn dieser Job ausgeführt wird. Um die Profiling-Daten anzuzeigen, müssen Sie diese Option aktivieren. Weitere Informationen über das Aktivieren und Anzeigen von Metriken finden Sie unter Auftragsüberwachung und Debugging.

Kennzahlen zur Jobbeobachtbarkeit

Schalten Sie die Erstellung zusätzlicher CloudWatch Messwerte für die Beobachtbarkeit ein, wenn dieser Job ausgeführt wird. Weitere Informationen finden Sie unter the section called “Überwachen mit AWS Glue Beobachtbarkeitsmetriken”.

Kontinuierliche Protokollierung

Aktivieren Sie die kontinuierliche Protokollierung bei Amazon CloudWatch. Wenn diese Option nicht aktiviert ist, stehen Protokolle erst nach Abschluss des Auftrags zur Verfügung. Weitere Informationen finden Sie unter the section called “Protokollierung für AWS Glue Jobs”.

Spark-Benutzeroberfläche

Aktivieren Sie die Verwendung der Spark-Benutzeroberfläche, um diese Aufgabe zu überwachen. Weitere Informationen finden Sie unter Aktivierung der Apache Spark-Weboberfläche für AWS Glue jobs.

Pfad der Spark-Benutzeroberfläche protokolliert

Der Pfad zum Schreiben von Protokollen, wenn die Spark-Benutzeroberfläche aktiviert ist.

Konfiguration der Protokollierung und Überwachung der Spark-Benutzeroberfläche

Wählen Sie eine der folgenden Optionen:

- Standard: Schreiben Sie Protokolle mit der AWS Glue Job-Run-ID als Dateinamen. Schalten Sie die Spark-UI-Überwachung in der AWS Glue Konsole ein.
- Legacy: Schreiben Sie Logs mit 'spark-application- {timestamp} 'als Dateinamen. Schalten Sie die Spark-UI-Überwachung nicht ein.
- Standard und Legacy: Schreiben Sie Protokolle sowohl an den Standard- als auch an den Legacy-Speicherort. Aktivieren Sie die Spark-UI-Überwachung in der AWS Glue Konsole.

Maximale Gleichzeitigkeit

Legt die maximale Anzahl gleichzeitiger Ausführungen fest, die für diese Ausführung zulässig sind. Der Standardwert ist 1. Bei Erreichen dieser Schwelle wird ein Fehler zurückgegeben. Der Höchstwert, den Sie angeben können, wird durch ein Service Limit gesteuert. Wenn beispielsweise beim Starten einer neuen Instance eine vorherige Ausführung läuft, möchten Sie einen Fehler zurückgeben, um zu verhindern, dass zwei Instance desselben Auftrags gleichzeitig ausgeführt werden.

Temporärer Pfad

Geben Sie den Speicherort eines Arbeitsverzeichnisses in Amazon S3 an, in das temporäre Zwischenergebnisse geschrieben werden, wenn das Skript AWS Glue ausgeführt wird. Vergewissern Sie sich, dass sich keine Datei mit dem Namen des temporären Verzeichnis im Pfad befindet. Dieses Verzeichnis wird beim AWS Glue Lesen und Schreiben in Amazon Redshift und bei bestimmten AWS Glue Transformationen verwendet.

> **Note**
>
> AWS Glue erstellt einen temporären Bucket für Jobs, falls in einer Region noch kein Bucket vorhanden ist. Dieser Bucket erlaubt möglicherweise den öffentlichen

Zugriff. Sie können entweder den Bucket in Amazon S3 ändern, um den öffentlichen Zugriffsblock festzulegen, oder den Bucket später löschen, nachdem alle Aufträge in dieser Region abgeschlossen sind.

Schwellenwert für die Verzögerungsbenachrichtigung (Minuten)

Legt den Schwellenwert (in Minuten) fest, bevor eine Verzögerungsbenachrichtigung gesendet wird. Sie können diesen Schwellenwert festlegen, um Benachrichtigungen zu senden, wenn ein `RUNNING`, `STARTING` oder `STOPPING` Auftragslauf mehr als eine erwartete Anzahl von Minuten andauert.

Security configuration (Sicherheitskonfiguration)

Wählen Sie eine Sicherheitskonfiguration aus der Liste aus. Eine Sicherheitskonfiguration gibt an, wie die Daten am Amazon-S3-Ziel verschlüsselt werden: keine Verschlüsselung, serverseitige Verschlüsselung mit von AWS KMS verwalteten Schlüsseln (SSE-KMS) oder mit von Amazon S3 verwalteten Verschlüsselungsschlüsseln (SSE-S3).

Server-side encryption

Wenn Sie diese Option wählen, werden die Daten beim Schreiben des ETL-Auftrags nach Amazon S3 im Ruhezustand mit SSE-S3-Verschlüsselung verschlüsselt. Sowohl Ihr Amazon-S3-Datenziel als auch alle Daten, die in ein temporäres Amazon-S3-Verzeichnis geschrieben werden, werden verschlüsselt. Diese Option wird als Auftragsparameter übergeben. Weitere Informationen finden Sie unter Schützen von Daten mit serverseitiger Verschlüsselung mit Amazon S3-verwalteten Verschlüsselungsschlüsseln (SSE-S3) im Benutzerhandbuch von Amazon Simple Storage Service.

> **Important**
>
> Diese Option wird bei Angabe einer Sicherheitskonfiguration ignoriert.

Verwenden des Glue-Datenkatalogs als Hive-Metastore

Wählen Sie diese Option aus, um den AWS Glue Datenkatalog als Hive-Metastore zu verwenden. Die für den Auftrag verwendete IAM-Rolle muss über die `glue:CreateDatabase`-Berechtigung verfügen. Sofern nicht bereits vorhanden, wird im Data Catalog eine Datenbank namens „default“ angelegt.

Verbindungen

Wählen Sie eine VPC-Konfiguration für den Zugriff auf Amazon S3 S3-Datenquellen in Ihrer Virtual Private Cloud (VPC). Sie können eine Netzwerkverbindung in erstellen und verwalten. AWS Glue Weitere Informationen finden Sie unter Herstellen einer Verbindung zu Daten.

Bibliotheken

Python-Bibliothekspfad, abhängiger JARs Pfad und Pfad der referenzierten Dateien

Geben Sie diese Optionen an, wenn Ihr Skript sie benötigt. Sie können die durch Kommas getrennten Amazon-S3-Pfade für diese Optionen definieren, wenn Sie den Auftrag definieren. Sie können diese Pfade bei der Ausführung des Auftrag überschreiben. Weitere Informationen finden Sie unter Bereitstellen eigener, benutzerdefinierter Skripts.

Auftragsparameter

Ein Satz von Schlüssel-Wert-Paaren, die als benannte Parameter an das Skript übergeben werden. Dies sind Standardwerte, die verwendet werden, wenn das Skript ausgeführt wird. Sie können auch in Auslösern oder beim Ausführen der Aufgabe überschrieben werden. Sie müssen dem Schlüsselnamen `--` voranstellen; z. B.: `--myKey`. Sie übergeben Jobparameter als Map, wenn Sie den verwenden AWS Command Line Interface.

Beispiele finden Sie unter „Python parameters“ (Python-Parameter) in Python-Parameter übergeben und darauf zugreifen in AWS Glue.

Tags

Markieren Sie Ihren Auftrag mit einem Tag-Schlüssel und einem optionalen Tag-Wert. Nachdem Tag-Schlüssel erstellt wurden, sind sie schreibgeschützt. Verwenden Sie Tags für manche Ressourcen, damit sie leichter zu organisieren und identifizieren sind. Weitere Informationen finden Sie unter AWS Schlagworte in AWS Glue.

Einschränkungen für Aufträge, die auf verwaltete Tabellen von Lake Formation zugreifen

Beachten Sie die folgenden Hinweise und Einschränkungen, wenn Sie Jobs erstellen, die aus Tabellen lesen oder in Tabellen schreiben, die von verwaltet werden AWS Lake Formation:

- Die folgenden Features werden in Aufträgen, die auf Tabellen mit Filtern auf Zellebene zugreifen, nicht unterstützt:
  - Auftrags-Lesezeichen und Begrenzte Ausführung

- Pushdown-Prädikate
- Prädikate für Serverseitige Katalogpartitionen
- enableUpdateCatalog

Spark-Skripte in der AWS Glue-Konsole bearbeiten

Ein Skript enthält den Code, der Daten aus Quellen extrahiert, transformiert und in Ziele lädt. AWS Glue führt ein Skript aus, wenn es einen Job startet.

ETL-Skripts in AWS Glue können in Python oder Scala codiert werden. Python-Skripte verwenden eine Sprache, die eine Erweiterung des PySpark Python-Dialekts für Extraktions-, Transformations- und Ladeaufträge (ETL) darstellt. Das Skript enthält erweiterte Konstrukte für die Verarbeitung von ETL-Transformationen. Wenn Sie die Quellcodelogik für Ihren Auftrag automatisch generieren, wird ein Skript erstellt. Sie können dieses Skript bearbeiten oder Ihr eigenes Skript zur Verarbeitung Ihrer ETL-Vorgänge bereitstellen.

Informationen zum Definieren und Bearbeiten von Skripts in AWS Glue finden Sie unter AWS Glue Leitfaden zur Programmierung.

Zusätzliche Bibliotheken oder Dateien

Wenn Ihr Skript zusätzliche Bibliotheken oder Dateien erfordert, können Sie sie wie folgt angeben:

Python-Bibliothekspfad

Durch Komma getrennte Amazon Simple Storage Service (Amazon S3)-Pfade zu den Python-Bibliotheken, die das Skript erfordert.

> **Note**
>
> Es können nur reine Python-Bibliotheken verwendet werden. Bibliotheken, die auf C-Erweiterungen basieren, wie zum Beispiel die Python Data Analysis Library von Panda , werden noch nicht unterstützt.

Abhängiger Jars-Pfad

Durch Komma getrennte Amazon-S3-Pfade zu JAR-Dateien, die vom Skript benötigt werden.

> **Note**
>
> Derzeit können nur reine Java- oder Scala (2.11)-Bibliotheken verwendet werden.

Pfad für referenzierte Dateien

Durch Komma getrennte Amazon-S3-Pfade zu zusätzlichen Dateien (z. B. Konfigurationsdateien), die das Skript erfordert.

Aufgräge (veraltet)

Ein Skript enthält den Code, der das Extrahieren, Transformieren und Laden (ETL) ausführt. Sie können Ihr eigenes Skript bereitstellen, oder AWS Glue kann unter Anleitung von Ihnen ein Skript generieren. Informationen zum Erstellen eigener Skripts finden Sie unter Bereitstellen eigener, benutzerdefinierter Skripts.

Sie können ein Skript bearbeiten in AWS Glue console. Wenn Sie ein Skript bearbeiten, können Sie Quellen, Ziele und Transformationen hinzufügen.

So bearbeiten Sie ein Skript

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/. Wählen Sie anschließend die Registerkarte Jobs (Aufträge) aus.
2. Wählen Sie einen Auftrag in der Liste aus, und wählen Sie dann Action (Aktion), Edit script (Skript bearbeiten) aus, um den Skript-Editor zu öffnen.

   Sie können den Skript-Editor auch von der Auftrags-Detailseite aus aufrufen. Wählen Sie die Registerkarte Script (Skript) und anschließend Edit script (Skript bearbeiten) aus.

Skript-Editor

Das Tool AWS Glue Mit dem Skripteditor können Sie Quellen, Ziele und Transformationen in Ihr Skript einfügen, ändern und löschen. Der Skript-Editor zeigt sowohl das Skript als auch ein Diagramm an, um Ihnen die Visualisierung des Datenflusses zu erleichtern.

Um ein Diagramm für das Skript zu erstellen, wählen Sie Diagramm generieren. AWS Glue verwendet Kommentarzeilen im Skript, die mit ## beginnen, um das Diagramm zu rendern. Um Ihr Skript im Diagramm korrekt darzustellen, müssen Sie die Parameter in den Kommentaren und die Parameter im Apache-Spark-Code synchron halten.

Mit dem Skript-Editor können Sie dort Code-Vorlagen hinzufügen, wo sich der Cursor im Skript befindet. Wählen Sie im oberen Bereich des Editors eine der folgenden Optionen aus:

- Um dem Skript eine Quelltabelle hinzuzufügen, wählen Sie Source (Quelle) aus.
- Um dem Skript eine Zieltabelle hinzuzufügen, wählen Sie Target (Ziel) aus.
- Um dem Skript einen Zielort hinzuzufügen, wählen Sie Target location (Zielort) aus.
- Um dem Skript eine Transformation hinzuzufügen, wählen Sie Transform (Transformieren) aus. Weitere Informationen über die Funktionen, die in Ihrem Skript aufgerufen werden, finden Sie unter Programmieren AWS Sie Glue ETL-Skripte in PySpark.
- Um dem Skript eine Spigot-Transformation hinzuzufügen, wählen Sie Spigot aus.

Ändern Sie im eingefügten Code die `parameters` sowohl in den Kommentaren als auch im Apache-Spark-Code. Wenn Sie z. B. eine Spigot-Transformation hinzufügen, vergewissern Sie sich, dass der `path` sowohl in der `@args` Kommentarzeile als auch in der `output` Codezeile ersetzt wird.

Auf der Registerkarte Logs (Protokolle) werden die Protokolle angezeigt, die Ihrem Auftrag zugeordnet sind, während er ausgeführt wird. Es werden die letzten 1.000 Zeilen angezeigt.

Auf der Registerkarte Schema wird das Schema der ausgewählten Quellen und Ziele angezeigt, sofern es in Data Catalog verfügbar ist.

Verfolgen von verarbeiteten Daten mit Auftragslesezeichen

AWS Glue verfolgt Daten, die bereits während einer vorherigen Ausführung eines ETL-Jobs verarbeitet wurden, indem Statusinformationen aus der Auftragsausführung beibehalten werden. Diese persistierende Zustandsinformation wird als Auftrags-Lesezeichen bezeichnet. Job-Lesezeichen helfen AWS Glue Behalten Sie die Statusinformationen bei und verhindern Sie die Wiederverarbeitung alter Daten. Mit Auftragslesezeichen können Sie neue Daten verarbeiten, wenn sie in einem geplanten Intervall erneut ausgeführt werden. Ein Aufgabenlesezeichen besteht aus den Zuständen für verschiedene Elemente von Aufgaben wie Quellen, Transformationen und Zielen. Ihr ETL-Job könnte beispielsweise neue Partitionen in einer Amazon S3 S3-Datei lesen. AWS Glue verfolgt, welche Partitionen der Job erfolgreich verarbeitet hat, um doppelte Verarbeitung und doppelte Daten im Zieldatenspeicher des Jobs zu verhindern.

Aufgabenlesezeichen werden für JDBC-Datenquellen, die Relationalize-Transformation und einige Amazon Simple Storage Service (Amazon S3)-Quellen implementiert. In der folgenden Tabelle sind die Amazon S3 S3-Quellformate aufgeführt, die AWS Glue unterstützt Job-Lesezeichen.

AWS Glue version	Quellformate für Amazon S3
Version 0.9	JSON, CSV, Apache Avro, XML
Version 1.0 und höher	JSON, CSV, Apache Avro, XML, Parquet, ORC

Für Informationen über AWS Glue Versionen finden Sie unterDefinieren von Auftragseigenschaften für Spark-Aufträge.

Die Funktion „Job-Lesezeichen“ bietet zusätzliche Funktionen, wenn sie über AWS Glue Skripts aufgerufen wird. Wenn Sie Ihr generiertes Skript durchsuchen, finden Sie möglicherweise Transformationskontexte, die sich auf dieses Feature beziehen. Weitere Informationen finden Sie unter the section called “Verwenden von Auftragslesezeichen”.

Themen

- Verwenden von Job-Lesezeichen in AWS Glue
- Operative Details des Features Auftragslesezeichen

Verwenden von Job-Lesezeichen in AWS Glue

Beim Start der Aufgabe wird die Aufgabenlesezeichenoption als Parameter übergeben. In der folgenden Tabelle werden die Optionen für das Setzen von Job-Lesezeichen auf dem beschrieben AWS Glue console.

Auftrags-Lesezeichen	Beschreibung
Aktivieren	Veranlasst den Auftrag, den Status nach einer Ausführung zu aktualisieren, um die zuvor verarbeiteten Daten zu verfolgen. Wenn Ihr Auftrag eine Quelle mit Unterstützung für Auftragslesezeichen hat, verfolgt er die verarbeiteten Daten, und wenn ein Auftrag ausgeführt wird, verarbeitet er neue Daten seit dem letzten Kontrollpunkt.

Auftrags-Lesezeichen	Beschreibung
Deaktivieren	Auftragslesezeichen werden nicht verwendet, und der Auftrag verarbeitet immer den gesamten Datensatz. Sie sind für die Verwaltung der Ausgaben aus früheren Auftragsausführungen verantwortlich. Dies ist die Standardeinstellung.
Pause	Verarbeiten Sie inkrementelle Daten seit der letzten erfolgreichen Ausführung oder die Daten in dem durch die folgenden Unteroptionen identifizierten Bereich, ohne den Status des letzten Lesezeichens zu aktualisieren. Sie sind für die Verwaltung der Ausgaben aus früheren Auftragsausführungen verantwortlich. Die beiden Unteroptionen sind: • job-bookmark-from<from-value>ist die Run-ID, die alle Eingaben darstellt, die bis zum letzten erfolgreichen Lauf verarbeitet wurden, einschließlich der angegebenen Run-ID. Die entsprechende Eingabe wird ignoriert. • job-bookmark-to<to-value>ist die Run-ID, die alle Eingaben darstellt , die bis zum letzten erfolgreichen Lauf verarbeitet wurden, einschlie ßlich der angegebenen Run-ID. Die entsprechende Eingabe ohne die Eingabe, die durch den <from-value> identifiziert wird, wird vom Auftrag verarbeitet. Jede Eingabe, die später als diese Eingabe ist, wird auch aus der Verarbeitung ausgeschlossen. Der Auftrags-Lesezeichenstatus wird nicht aktualisiert, wenn dieser Optionssatz angegeben wird. Die Unteroptionen sind optional, aber wenn sie verwendet werden, müssen beide Unteroptionen bereitgestellt werden.

Weitere Details zu den Parametern, die an Aufgaben in der Befehlszeile übergeben werden, und insbesondere zu Aufgabenlesezeichen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Für Amazon S3 S3-Eingangsquellen AWS Glue Job-Lesezeichen überprüfen den Zeitpunkt der letzten Änderung der Objekte, um zu überprüfen, welche Objekte erneut verarbeitet werden müssen.

Wenn Ihre Eingabedaten seit der letzten Auftragsausführung geändert wurden, werden die Dateien bei der erneuten Ausführung des Auftrags erneut verarbeitet.

Für JDBC-Quellen gelten die folgenden Regeln:

- Für jede Tabelle AWS Glue verwendet eine oder mehrere Spalten als Lesezeichenschlüssel, um neue und verarbeitete Daten zu ermitteln. Die Lesezeichen-Tasten werden zu einem einzigen zusammengesetzten Schlüssel zusammengefasst.
- AWS Glue verwendet standardmäßig den Primärschlüssel als Lesezeichenschlüssel, vorausgesetzt, er nimmt sequentiell zu oder ab (ohne Lücken).
- Sie können die Spalten angeben, die als Lesezeichenschlüssel in Ihrem Skript verwendet werden sollen. AWS Glue Weitere Hinweise zur Verwendung von Job-Lesezeichen in AWS Glue Skripten finden Sie unterthe section called “Verwenden von Auftragslesezeichen”.
- AWS Glue unterstützt nicht die Verwendung von Spalten mit Namen, bei denen Groß- und Kleinschreibung beachtet wird, als Schlüssel für Job-Lesezeichen.

Sie können Ihre Job-Lesezeichen für Ihre zurückspulen AWS Glue Fügen Sie ETL-Jobs zu einer beliebigen vorherigen Jobausführung hinzu. Mit der Funktion werden Datenauffüllszenarien vereinfacht. Sie können Auftragslesezeichen zu jeder beliebigen früheren Auftragsausführung zurückspulen, was dazu führt, dass in der nachfolgenden Auftragsausführung nur die Daten aus der mit Lesezeichen versehenen Auftragsausführung erneut verarbeitet werden.

Wenn Sie beabsichtigen, alle Daten mit demselben Auftrag erneut zu verarbeiten, setzen Sie das Auftragslesezeichen zurück. Um den Status des Job-Lesezeichens zurückzusetzen, verwenden Sie AWS Glue Konsole, die ResetJobBookmark Aktion (Python: reset_job_bookmark) API-Operation oder die AWS CLI. Geben Sie z. B. den folgenden Befehl mit der AWS CLI ein:

```
aws glue reset-job-bookmark --job-name my-job-name
```

Wenn Sie ein Lesezeichen zurückspulen oder zurücksetzen, AWS Glue bereinigt die Zieldateien nicht, da es mehrere Ziele geben kann und Ziele nicht mit Job-Lesezeichen verfolgt werden. Nur Quelldateien werden mit Auftragslesezeichen verfolgt. Beim Zurückspulen und erneuten Verarbeiten der Quelldateien können Sie verschiedene Ausgabeziele erstellen, um doppelte Daten in der Ausgabe zu vermeiden.

AWS Glue verfolgt die Job-Lesezeichen für jeden Job. Wenn Sie einen Auftrag löschen, wird das Auftrags-Lesezeichen gelöscht.

In einigen Fällen haben Sie möglicherweise aktiviert AWS Glue Job-Lesezeichen, aber Ihr ETL-Job verarbeitet Daten erneut, die bereits in einem früheren Lauf verarbeitet wurden. Informationen zur Behebung häufiger Ursachen für diesen Fehler finden Sie unter Fehler beheben Spark-Fehler.

Operative Details des Features Auftragslesezeichen

Dieser Abschnitt beschreibt weitere operative Details zur Verwendung von Auftragslesezeichen.

Auftragslesezeichen speichern die Zustände für einen Auftrag. Jede Instance des Status wird durch einen Auftragsnamen und eine Versionsnummer gekennzeichnet. Wenn ein Skript `job.init` aufruft, ruft es seinen Zustand ab und erhält immer die neueste Version. Innerhalb eines Zustands gibt es mehrere Zustandselemente, die für jede Quell-, Transformations- und Speicher-Instance im Skript spezifisch sind. Diese Zustandselemente werden durch einen Transformationskontext identifiziert, der an das entsprechende Element (Quelle, Transformation oder Speicher) im Skript angefügt ist. Die Zustandselemente werden atomar gespeichert, wenn `job.commit` aus dem Benutzerskript aufgerufen wird. Das Skript ruft den Auftragsnamen und die Steuerungsoption für die Auftragslesezeichen aus den Argumenten ab.

Die Zustandselemente im Auftragslesezeichen sind quell-, transformations- oder speicherspezifische Daten. Angenommen, Sie möchten beispielsweise inkrementelle Daten aus einem Amazon-S3-Speicherort, an den ständig von einem vorgelagerten Auftrag oder Prozess geschrieben wird, lesen. In diesem Fall muss das Skript feststellen, was bisher verarbeitet wurde. Die Implementierung des Auftragslesezeichens für die Amazon-S3-Quelle speichert Informationen, so dass der Auftrag beim erneuten Ausführen nur die neuen Objekte anhand der gespeicherten Informationen filtern und den Status für den nächsten Durchlauf des Auftrags neu berechnen muss. Ein Zeitstempel wird verwendet, um die neuen Dateien zu filtern.

Zusätzlich zu den Zustandselementen haben Auftragslesezeichen eine Laufnummer, eine Versuchsnummer und eine Versionsnummer. Die Laufnummer verfolgt den Lauf des Auftrags, und die Versuchsnummer zeichnet die Versuche für einen Auftragslauf auf. Die Auftragslaufnummer ist eine monoton steigende Zahl, die bei jedem erfolgreichen Lauf erhöht wird. Die Versuchsnummer verfolgt die Versuche für jeden Lauf und wird nur dann erhöht, wenn es nach einem fehlgeschlagenen Versuch einen Lauf gibt. Die Versionsnummer steigt monoton an und verfolgt die Updates eines Auftragslesezeichens.

In der AWS Glue Servicedatenbank werden die Lesezeichenstatus für alle Transformationen zusammen als Schlüssel-Wert-Paare gespeichert:

```
{
  "job_name" : ...,
  "run_id": ...,
  "run_number": ..,
  "attempt_number": ...
  "states": {
    "transformation_ctx1" : {
      bookmark_state1
    },
    "transformation_ctx2" : {
      bookmark_state2
    }
  }
}
```

Bewährte Methoden

Im Folgenden finden Sie bewährte Methoden zur Nutzung von Auftragslesezeichen.

- Ändern Sie die Datenquelleneigenschaft nicht, wenn das Lesezeichen aktiviert ist. Beispielsweise gibt es eine Datenquelle „datasource0“, die auf einen Eingabepfad A von Amazon S3 verweist, und der Auftrag hat aus einer Quelle gelesen, die mehrere Runden lang mit aktiviertem Lesezeichen ausgeführt wurde. Wenn Sie den Eingabepfad von datasource0 in Amazon S3-Pfad B ändern, ohne das zu ändern`transformation_ctx`, verwendet der AWS Glue Job den alten gespeicherten Lesezeichenstatus. Dies führt dazu, dass Dateien im Eingabepfad B fehlen oder übersprungen werden, da AWS Glue davon ausgegangen wird, dass diese Dateien in früheren Durchläufen verarbeitet wurden.
- Verwenden Sie eine Katalogtabelle mit Lesezeichen für eine bessere Partitionsverwaltung. Lesezeichen funktionieren sowohl für Datenquellen aus dem Datenkatalog als auch aus Optionen. Es ist jedoch schwierig, neue Partitionen aus Optionen heraus zu entfernen/hinzuzufügen. Die Verwendung einer Katalogtabelle mit Crawlern kann eine bessere Automatisierung bieten, um die neu hinzugefügten Partitionen nachzuverfolgen, und gibt Ihnen die Flexibilität zur Auswahl bestimmter Partitionen mit einem Pushdown-Prädikat.
- Verwenden Sie die Dateiauflistung von AWS Glue für Amazon S3 für große Datensätze. Ein Lesezeichen listet alle Dateien unter jeder Eingabepartition auf und filtert sie. Wenn sich also zu viele Dateien unter einer einzelnen Partition befinden, kann das Lesezeichen den OOM-Fehler für den Treiber auslösen. Verwenden Sie den AWS Glue Amazon S3 S3-Datei-Lister, um zu vermeiden, dass alle Dateien im Speicher gleichzeitig aufgelistet werden.

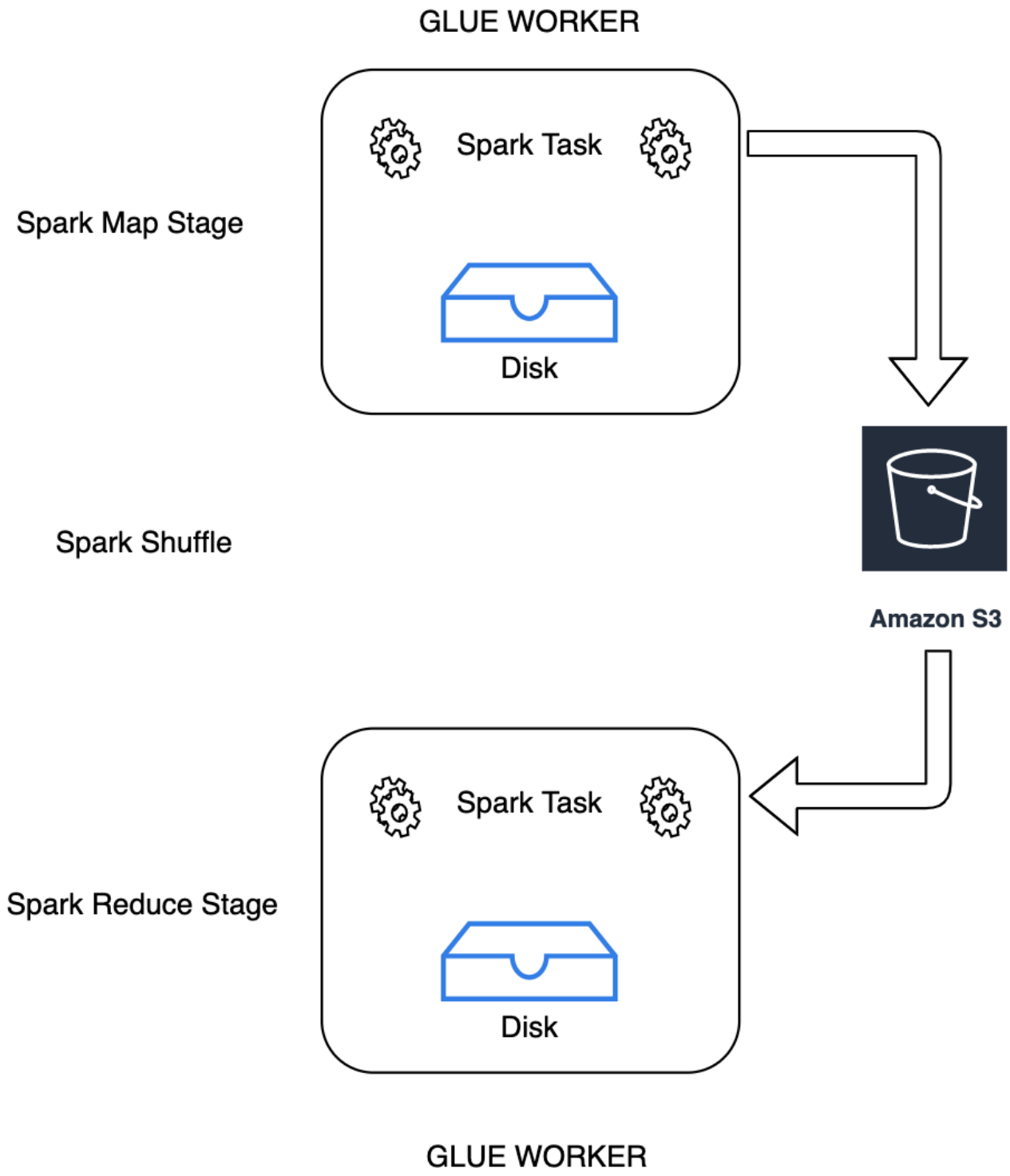
Speichern von Spark-Shuffle-Daten

Shuffling ist ein wichtiger Schritt in einem Spark-Job, wenn Daten zwischen Partitionen neu angeordnet werden. Dies ist erforderlich, da umfassende Transformationen wie `join`, `groupByKey`, `reduceByKey` und `repartition` Informationen von anderen Partitionen benötigen, um die Verarbeitung abzuschließen. Spark sammelt die erforderlichen Daten von jeder Partition und kombiniert sie zu einer neuen Partition. Beim Mischen werden Daten auf die Festplatte geschrieben und über das Netzwerk übertragen. Infolgedessen ist der Shuffle-Vorgang an die lokale Festplattenkapazität gebunden. Spark gibt den Fehler `No space left on device` oder den Fehler `MetadataFetchFailedException` aus, wenn auf dem Executor nicht genügend Speicherplatz übrig ist und keine Wiederherstellung vorliegt.

> **Note**
>
> AWS Glue Das Spark-Shuffle-Plugin mit Amazon S3 wird nur für AWS Glue ETL-Jobs unterstützt.

Lösung

Mit AWS Glue, können Sie jetzt Amazon S3 verwenden, um Spark-Shuffle-Daten zu speichern. Amazon S3 ist ein Objektspeicherservice, der branchenführende Skalierbarkeit, Datenverfügbarkeit, Sicherheit und Leistung bietet. Diese Lösung teilt Rechenleistung und Speicher für Ihre Spark-Aufträge auf und bietet vollständige Elastizität und kostengünstigen Shuffle-Speicher, sodass Sie Ihre Shuffle-intensiven Workloads zuverlässig ausführen können.



Wir stellen ein neues Cloud-Shuffle-Speicher-Plugin für Apache Spark zur Verwendung von Amazon S3 vor. Sie können Amazon S3 Shuffling einschalten, um Ihre AWS Glue Jobs zuverlässig und ohne Ausfälle, wenn bekannt ist, dass sie bei großen Shuffle-Vorgängen an die lokale Festplattenkapazität gebunden sind. In einigen Fällen ist das Shuffling zu Amazon S3 geringfügig langsamer als die lokale

Festplatte (oder EBS), wenn Sie eine große Anzahl kleiner Partitionen oder Shuffle-Dateien haben, die in Amazon S3 geschrieben wurden.

Voraussetzungen für die Verwendung des Cloud-Shuffle-Speicher-Plugins

Um das Cloud Shuffle Storage Plugin mit AWS Glue ETL-Jobs verwenden zu können, benötigen Sie Folgendes:

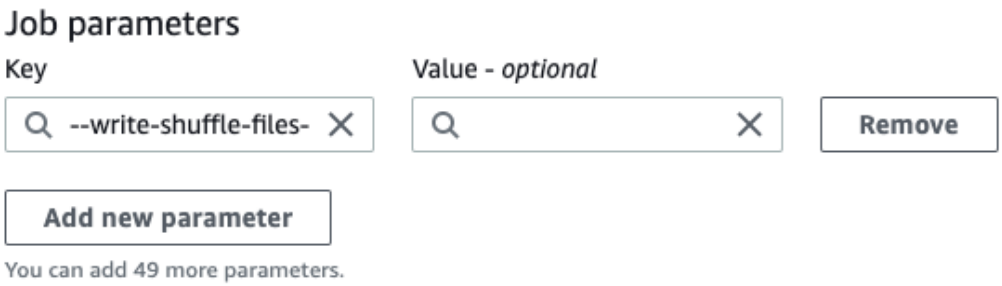
- Ein Amazon-S3-Bucket, der sich in der gleichen Region befindet, in der Ihr Auftrag ausgeführt wird, um die Shuffle- und Ausgabedaten zwischenzuspeichern. Das Amazon-S3-Präfix des Shuffle-Speichers kann wie im folgenden Beispiel mit `--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket`*`/`*`prefix`*`/` angegeben werden:

  ```
  --conf spark.shuffle.glue.s3ShuffleBucket=s3://glue-shuffle-123456789-us-east-1/glue-shuffle-data/
  ```

- Legen Sie die Lebenszyklusrichtlinien für den Amazon-S3-Speicher auf das Präfix fest (z. B. `glue-shuffle-data`), da der Shuffle-Manager die Dateien nach Abschluss der Aufgabe nicht bereinigt. Die dazwischen liegenden Shuffle- und Ausgabedaten sollten nach Beendigung eines Auftrags gelöscht werden. Benutzer können Richtlinien für einen kurzen Lebenszyklus für das Präfix festlegen. Anweisungen zum Einrichten einer Amazon-S3-Lebenszyklusrichtlinie finden Sie unter Lebenszykluskonfiguration für einen Bucket festlegen im Benutzerhandbuch für Amazon Simple Storage Service.

Die Verwendung von AWS Glue Spark Shuffle Manager von der Konsole aus AWS

Um das einzurichten AWS Glue Spark Shuffle Manager mit dem AWS Glue Konsole oder AWS Glue Studio bei der Konfiguration eines Jobs: Wählen Sie den Job-Parameter -- write-shuffle-files-to -s3, um Amazon S3 S3-Shuffling für den Job zu aktivieren.



Die Verwendung von AWS Glue Spark-Shuffle-Plugin

Die folgenden Jobparameter schalten den ein und stimmen ihn ab AWS Glue Shuffle-Manager. Da es sich bei diesen Parametern um Flags handelt, werden die angegebenen Werte nicht berücksichtigt.

- `--write-shuffle-files-to-s3`— Die Hauptflagge, die aktiviert AWS Glue Spark Shuffle Manager zur Verwendung von Amazon S3 S3-Buckets zum Schreiben und Lesen von Shuffle-Daten. Wenn das Flag nicht angegeben ist, wird der Shuffle-Manager nicht verwendet.
- `--write-shuffle-spills-to-s3`— (Wird nur unterstützt auf AWS Glue Version 2.0). Ein optionales Flag, mit dem Sie Ausgabeateien in Amazon-S3-Buckets auslagern können, was zusätzliche Ausfallsicherheit für Ihren Spark-Auftrag bereitstellt. Das ist nur bei großen Workloads erforderlich, bei denen viele Daten unabsichtlich auf der Festplatte landen. Wenn das Flag nicht angegeben ist, werden keine Ausgabe-Zwischendateien geschrieben.
- `--conf spark.shuffle.glue.s3ShuffleBucket=s3://<shuffle-bucket>` – Ein weiterer optionaler Parameter, der den Amazon S3 Bucket angibt, in den Sie die Shuffle-Dateien schreiben. Standardmäßig `--TempDir` /shuffle-data. AWS Glue 3.0+ unterstützt das Schreiben von Shuffle-Dateien in mehrere Buckets, indem Buckets mit einem Kommatrennzeichen angegeben werden, wie in. `--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket-1`*`/`*`prefix`*`,s3://`*`shuffle-bucket-2`*`/`*`prefix`*`/` Die Verwendung mehrerer Buckets verbessert die Leistung.

Sie müssen Sicherheitskonfigurationseinstellungen vornehmen, um die Verschlüsselung im Ruhezustand für die Shuffle-Daten zu aktivieren. Weitere Informationen zu den Sicherheitskonfigurationen finden Sie unter the section called “Einrichten der Verschlüsselung”. AWS Glue unterstützt alle anderen von Spark bereitgestellten Shuffle-bezogenen Konfigurationen.

Software-Binärdateien für das Cloud-Shuffle-Speicher-Plugin

Sie können auch die Software-Binärdateien des Cloud-Shuffle-Speicher-Plugins für Apache Spark unter der Apache-2.0-Lizenz herunterladen und in jeder Spark-Umgebung ausführen. Das neue Plugin bietet out-of-the Box-Unterstützung für Amazon S3 und kann auch einfach für die Verwendung anderer Formen von Cloud-Speicher wie Google Cloud Storage und Microsoft Azure Blob Storage konfiguriert werden. Weitere Informationen finden Sie unter Cloud-Shuffle-Speicher-Plugin für Apache Spark.

Hinweise und Einschränkungen

Im Folgenden finden Sie Hinweise oder Einschränkungen für AWS Glue Shuffle-Manager:

- AWS Glue Shuffle Manager löscht die (temporären) Shuffle-Datendateien, die in Ihrem Amazon S3 S3-Bucket gespeichert sind, nicht automatisch, nachdem ein Job abgeschlossen ist. Befolgen Sie zur Gewährleistung des Datenschutzes die Anweisungen in Voraussetzungen für die Verwendung des Cloud-Shuffle-Speicher-Plugins, bevor Sie das Cloud-Shuffle-Speicher-Plugin aktivieren.
- Sie können dieses Feature verwenden, wenn Ihre Daten verzerrt sind.

Cloud-Shuffle-Speicher-Plugin für Apache Spark

Das Cloud-Shuffle-Speicher-Plugin ist ein mit der `ShuffleDataIO`-API kompatibles Apache-Spark-Plugin, das das Speichern von Shuffle-Daten auf Cloud-Speichersystemen (wie Amazon S3) ermöglicht. Es unterstützt Sie bei der Ergänzung oder dem Ersatz lokaler Festplattenspeicherkapazität für große Shuffle-Operationen, die häufig durch Transformationen wie `join`, `reduceByKey`, `groupByKey` und `repartition` in Ihren Spark-Anwendungen ausgelöst werden. Dadurch werden häufige Ausfälle oder Preis-/Leistungsverschiebungen bei Ihren Serverless-Datenanalyseaufträgen und Pipelines reduziert.

AWS Glue

AWS Glue In den Versionen 3.0 und 4.0 ist das Plugin vorinstalliert und bereit, das Mischen zu Amazon S3 ohne zusätzliche Schritte zu ermöglichen. Weitere Informationen finden Sie unter AWS Glue Spark-Shuffle-Plugin mit Amazon S3, um die Funktion für Ihre Spark-Anwendungen zu aktivieren.

Andere Spark-Umgebungen

Für das Plugin müssen die folgenden Spark-Konfigurationen in anderen Spark-Umgebungen festgelegt werden:

- `--conf spark.shuffle.sort.io.plugin.class=com.amazonaws.spark.shuffle.io.cloud.Choppe`
  Dies informiert Spark, dieses Plugin für Shuffle IO zu verwenden.
- `--conf spark.shuffle.storage.path=s3://`*`bucket-name`*`/`*`shuffle-file-dir`*: Der Pfad, in dem Ihre Shuffle-Dateien gespeichert werden.

> **Note**
>
> Das Plugin überschreibt eine Spark-Kernklasse. Daher muss das Plugin-Jar vor den Spark-Jars geladen werden. Sie können dies `userClassPathFirst` in YARN-Umgebungen vor Ort tun, wenn das Plugin außerhalb verwendet wird AWS Glue.

Bündeln des Plugins mit Ihren Spark-Anwendungen

Sie können das Plugin mit Ihren Spark-Anwendungen und Spark-Verteilungen (Versionen 3.1 und höher) bündeln, indem Sie die Plugin-Abhängigkeit in Ihrem Maven `pom.xml` hinzufügen, während Sie Ihre Spark-Anwendungen lokal entwickeln. Weitere Informationen zu den Plugin- und Spark-Versionen finden Sie unter Plugin-Versionen.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
    </repository>
</repositories>
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>chopper-plugin</artifactId>
    <version>3.1-amzn-LATEST</version>
</dependency>
```

Sie können die Binärdateien alternativ direkt von AWS Glue Maven-Artefakten herunterladen und sie wie folgt in Ihre Spark-Anwendung einbinden.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
chopper-plugin/3.1-amzn-LATEST/chopper-plugin-3.1-amzn-LATEST.jar -P /usr/lib/spark/
jars/
```

Beispiel spark-submit

```
spark-submit --deploy-mode cluster \
--conf spark.shuffle.storage.s3.path=s3://<ShuffleBucket>/<shuffle-dir> \
```

```
--conf spark.driver.extraClassPath=<Path to plugin jar> \
--conf spark.executor.extraClassPath=<Path to plugin jar> \
--class <your test class name> s3://<ShuffleBucket>/<Your application jar> \
```

Optionale Konfigurationen

Dies sind optionale Konfigurationswerte, die das Amazon-S3-Zufallsverhalten steuern.

- `spark.shuffle.storage.s3.enableServerSideEncryption`: Aktivieren/Deaktivieren von S3 SSE für Shuffle- und Spill-Dateien. Der Standardwert ist `true`.
- `spark.shuffle.storage.s3.serverSideEncryption.algorithm`: Der zu verwendende SSE-Algorithmus. Der Standardwert ist `AES256`.
- `spark.shuffle.storage.s3.serverSideEncryption.kms.key`: Der KMS-Schlüssel-ARN, wenn SSE aws:kms aktiviert ist.

Neben diesen Konfigurationen müssen Sie möglicherweise Konfigurationen wie `spark.hadoop.fs.s3.enableServerSideEncryption` und andere umgebungsspezifische Konfigurationen festlegen, um sicherzustellen, dass die für Ihren Anwendungsfall geeignete Verschlüsselung angewendet wird.

Plugin-Versionen

Dieses Plugin wird für die Spark-Versionen unterstützt, die jeder AWS Glue Version zugeordnet sind. Die folgende Tabelle zeigt die AWS Glue Version, Spark-Version und die zugehörige Plugin-Version mit dem Amazon S3 S3-Speicherort für die Software-Binärdatei des Plugins.

AWS Glue version	Spark-Version	Plugin-Version	Amazon-S3-Speicherort
3.0	3.1	3.1-amzn-LATEST	s3://aws-glue-etl-artifacts/release/com/amazonaws/chopper-plugin/3.1-amzn-0/chopper-Plugin-3.1-amzn-latest.jar
4,0	3.3	3.3-amzn-LATEST	s3://aws-glue-etl-artifacts/release/com/

AWS Glue version	Spark-Version	Plugin-Version	Amazon-S3-Speicherort
			amazonaws/chopper-plugin/3.3-amzn-0/chopper-plugin-3.3-amzn-neuest.jar

License

Die Software-Binärdatei für dieses Plugin ist unter der Apache-2.0-Lizenz lizenziert.

Überwachen AWS Glue Stellen bei Spark

Themen

- Spark-Metriken sind verfügbar in AWS Glue Studio
- Überwachen von Aufgaben über die Apache-Spark-Webbenutzeroberfläche
- Überwachung mit Einblicken in die AWS Glue Auftragsausführung
- Überwachung mit Amazon CloudWatch
- Auftragsüberwachung und Debugging

Spark-Metriken sind verfügbar in AWS Glue Studio

Die Registerkarte Metrics (Metriken) zeigt Metriken, die erfasst werden, wenn ein Auftrag ausgeführt wird und das Profiling aktiviert ist. Die folgenden Diagramme werden in Spark-Aufträgen angezeigt:

- ETL-Datenbewegung
- Speicherprofil: Treiber und Executors

Wählen Sie View additional metrics (Weitere Metriken anzeigen), um die folgenden Graphen anzuzeigen:

- ETL-Datenbewegung
- Speicherprofil: Treiber und Executors
- Datenmischung über Executors hinweg

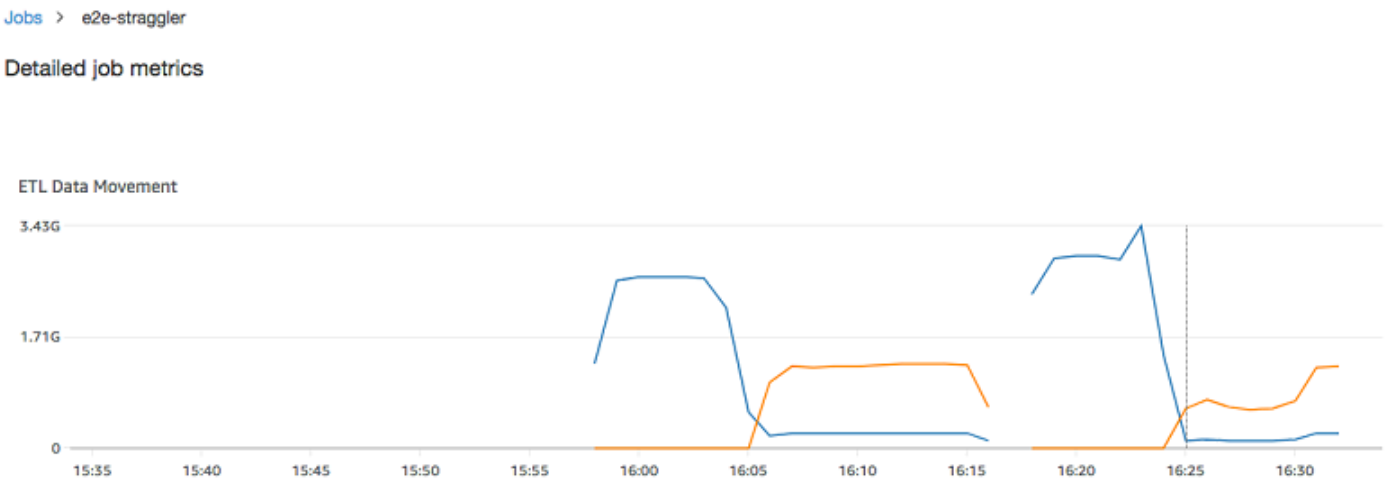
- CPU-Auslastung: Treiber und Executors
- Auftragsausführung: Aktive Executors, Abgeschlossen Phasen und maximal benötigte Executors

Die Daten für diese Diagramme werden in CloudWatch Metriken übertragen, wenn der Job für die Erfassung von Metriken konfiguriert ist. Weitere Informationen über das Aktivieren von Metriken und die Interpretation der Graphen finden Sie unter Auftragsüberwachung und Debugging.

Example Der Graph für die ETL-Datenbewegung

Der Graph für die ETL-Datenbewegung zeigt die folgenden Metriken:

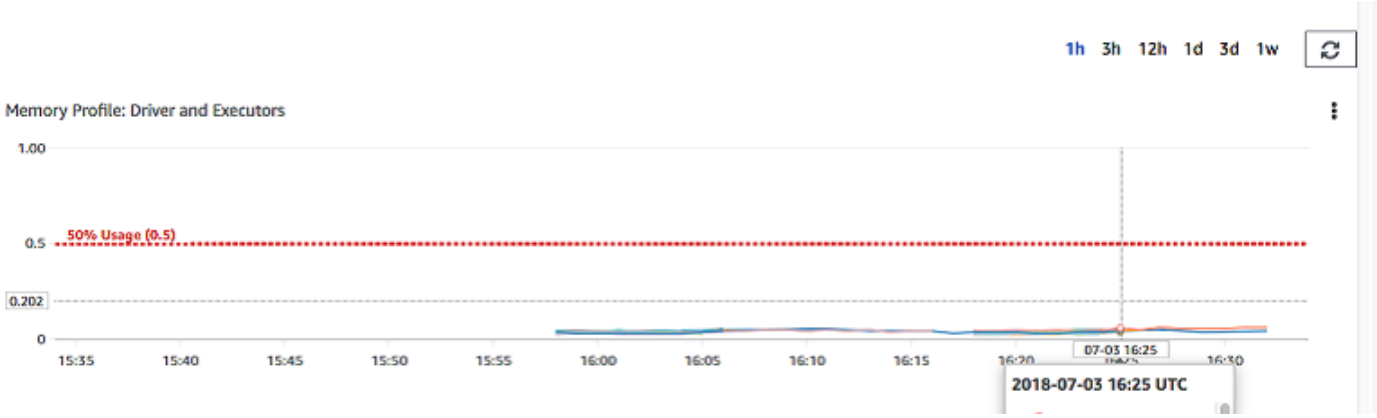
- Die Anzahl der von allen Executors von Amazon S3 gelesenen Bytes – `glue.ALL.s3.filesystem.read_bytes`
- Die Anzahl der von allen Executors in Amazon S3 geschriebenen Bytes – `glue.ALL.s3.filesystem.write_bytes`



Example Der Graph für das Speicherprofil

Der Graph für das Speicherprofil zeigt die folgenden Metriken:

- Der vom JVM-Heap für diesen Treiber (Skalierung: 0-1) verwendete Speicheranteil – vom Treiber, einem durch executorId identifizierten Executor oder allen Executors –
  - `glue.driver.jvm.heap.usage`
  - `glue.executorId.jvm.heap.usage`
  - `glue.ALL.jvm.heap.usage`



Example Der Graph für die Datenmischung über Executors hinweg

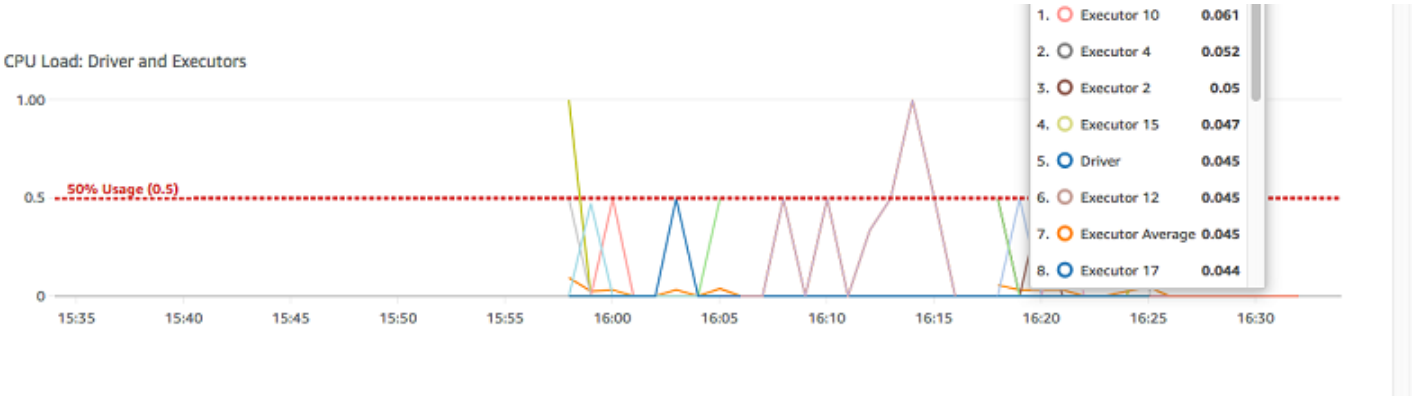
Der Graph für das Mischen der Daten über alle Executors hinweg zeigt die folgenden Metriken:

- Die Zahl der von allen Executors gelesenen Bytes, um ihre Daten zu mischen - `glue.driver.aggregate.shuffleLocalBytesRead`
- Die Zahl der von allen Executors geschriebenen Bytes, um ihre Daten zu mischen - `glue.driver.aggregate.shuffleBytesWritten`

Example Der Graph für die CPU-Auslastung

Der Graph für die CPU-Auslastung zeigt die folgenden Metriken:

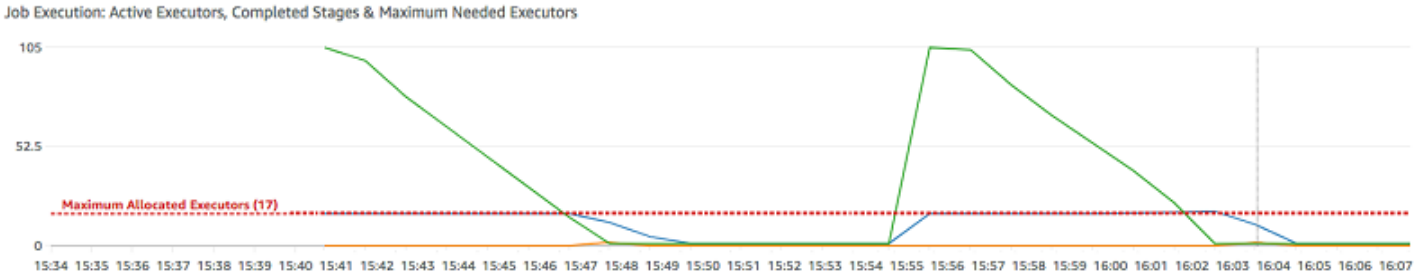
- Der vom Treiber verwendete Anteil der CPU-Systemauslastung (Skalierung: 0-1) – vom Treiber, einem durch executorId identifizierten Executor oder allen Executors –
  - `glue.driver.system.cpuSystemLoad`
  - `glue.`*`executorId`*`.system.cpuSystemLoad`
  - `glue.ALL.system.cpuSystemLoad`



Example Der Graph für die Auftragsausführung

Der Graph für die Auftragsausführung zeigt die folgenden Metriken:

- Die Anzahl der aktiven Executors - `glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`
- Die Anzahl der abgeschlossenen Phasen - `glue.aggregate.numCompletedStages`
- Die Anzahl maximal benötigter Executors - `glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`



Überwachen von Aufgaben über die Apache-Spark-Webbenutzeroberfläche

Sie können die Apache-Spark-Webbenutzeroberfläche zum Überwachen und Debuggen von AWS Glue-ETL-Aufträgen verwenden, die auf dem AWS Glue-Aufgabensystem ausgeführt werden, und von Spark-Anwendungen, die auf AWS Glue-Entwicklungsendpunkten ausgeführt werden. Sie können in der Spark-Benutzeroberfläche folgende Punkte für die einzelnen Aufgaben überprüfen:

- Ereignis-Zeitplan der einzelnen Spark-Phasen
- Ausgerichtetes azyklisches Diagramm (Directed Acyclic Graph, DAG) der Aufgabe
- Physische und logische Pläne für SparkSQL-Abfragen

- Zugrunde liegende Spark-Umgebungsvariablen für die einzelnen Aufgaben

Weitere Informationen zur Verwendung der Spark-Web-UI finden Sie unter Web-UI in der Spark-Dokumentation. Anleitungen dazu, wie Sie die Ergebnisse der Spark-Benutzeroberfläche interpretieren können, um die Leistung Ihres Jobs zu verbessern, finden Sie unter Bewährte Methoden zur Leistungsoptimierung AWS Glue für Apache Spark-Jobs in AWS Prescriptive Guidance.

Sie können die Spark-Benutzeroberfläche in der AWS Glue Konsole sehen. Dies ist verfügbar, wenn ein AWS Glue Job auf Versionen AWS Glue 3.0 oder höher ausgeführt wird, wobei die Logs im Standardformat (und nicht im Legacy-Format) generiert werden, was der Standard für neuere Jobs ist. Wenn Sie über Protokolldateien mit mehr als 0,5 GB verfügen, können Sie die Unterstützung für fortlaufende Protokolle für Auftragsausführungen in Versionen AWS Glue 4.0 oder höher aktivieren, um die Archivierung, Analyse und Problembehandlung von Protokollen zu vereinfachen.

Sie können die Spark-Benutzeroberfläche mithilfe der AWS Glue Konsole oder der AWS Command Line Interface (AWS CLI) aktivieren. Wenn Sie die Spark-Benutzeroberfläche aktivieren, können AWS Glue-ETL-Aufträge und Spark-Anwendungen in AWS Glue-Entwicklungsendpunkten Spark-Ereignisprotokolle an einem Speicherort sichern, den Sie in Amazon Simple Storage Service (Amazon S3) angeben. Die so in Amazon S3 gesicherten Ereignisprotokolle können in der Spark-Benutzeroberfläche in Echtzeit während der Auftragsausführung und nach Abschluss des Auftrags verwendet werden. Die Protokolle verbleiben zwar in Amazon S3, können aber über die Spark-Benutzeroberfläche in der AWS Glue Konsole angezeigt werden.

Berechtigungen

Um die Spark-Benutzeroberfläche in der AWS Glue Konsole zu verwenden, können Sie alle einzelnen Dienste verwenden `UseGlueStudio` oder hinzufügen APIs. Alle APIs sind erforderlich, um die Spark-Benutzeroberfläche vollständig nutzen zu können. Benutzer können jedoch auf die Funktionen von SparkUI zugreifen, indem sie den Dienst APIs zu ihrer IAM-Berechtigung hinzufügen, um einen detaillierten Zugriff zu erhalten.

`RequestLogParsing`ist am kritischsten, da es das Parsen von Logs durchführt. Die übrigen APIs dienen zum Lesen der jeweiligen analysierten Daten. `GetStages`Bietet beispielsweise Zugriff auf die Daten zu allen Phasen eines Spark-Jobs.

Die Liste der APIs zugewiesenen Spark-UI-Services finden `UseGlueStudio` Sie weiter unten in der Beispielrichtlinie. Die folgende Richtlinie bietet Zugriff auf die Nutzung nur der Funktionen der Spark-

Benutzeroberfläche. Informationen zum Hinzufügen weiterer Berechtigungen wie Amazon S3 und IAM finden Sie unter Erstellen benutzerdefinierter IAM-Richtlinien für. AWS Glue Studio

Die Liste der APIs zugewiesenen Spark-UI-Services finden Sie weiter unten in der Beispielrichtlinie. `UseGlueStudio` Wenn Sie eine Spark-UI-Service-API verwenden, verwenden Sie den folgenden Namespace:. `glue:<ServiceAPI>`

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowGlueStudioSparkUI",
   "Effect": "Allow",
            "Action": [
                "glue:RequestLogParsing",
                "glue:GetLogParsingStatus",
                "glue:GetEnvironment",
                "glue:GetJobs",
                "glue:GetJob",
                "glue:GetStage",
                "glue:GetStages",
                "glue:GetStageFiles",
                "glue:BatchGetStageFiles",
                "glue:GetStageAttempt",
                "glue:GetStageAttemptTaskList",
                "glue:GetStageAttemptTaskSummary",
                "glue:GetExecutors",
                "glue:GetExecutorsThreads",
                "glue:GetStorage",
                "glue:GetStorageUnit",
                "glue:GetQueries",
                "glue:GetQuery"
            ],
   "Resource": [
    "*"
   ]
  }
 ]
}
```

Einschränkungen

- Die Spark-Benutzeroberfläche in der AWS Glue Konsole ist für Jobausführungen, die vor dem 20. November 2023 stattfanden, nicht verfügbar, da sie im alten Protokollformat vorliegen.
- Die Spark-Benutzeroberfläche in der AWS Glue Konsole unterstützt fortlaufende Logs für AWS Glue 4.0, wie sie beispielsweise standardmäßig bei Streaming-Jobs generiert werden. Die maximale Summe aller generierten gerollten Protokollereignisdateien beträgt 2 GB. Für AWS Glue Jobs ohne Unterstützung für Rolling-Logs beträgt die maximale Größe der Protokollereignisdatei, die für SparkUI unterstützt wird, 0,5 GB.
- Die serverlose Spark-Benutzeroberfläche ist nicht für Spark-Ereignisprotokolle verfügbar, die in einem Amazon S3 S3-Bucket gespeichert sind und auf die nur Ihre VPC zugreifen kann.

Beispiel: Web-UI von Apache Spark

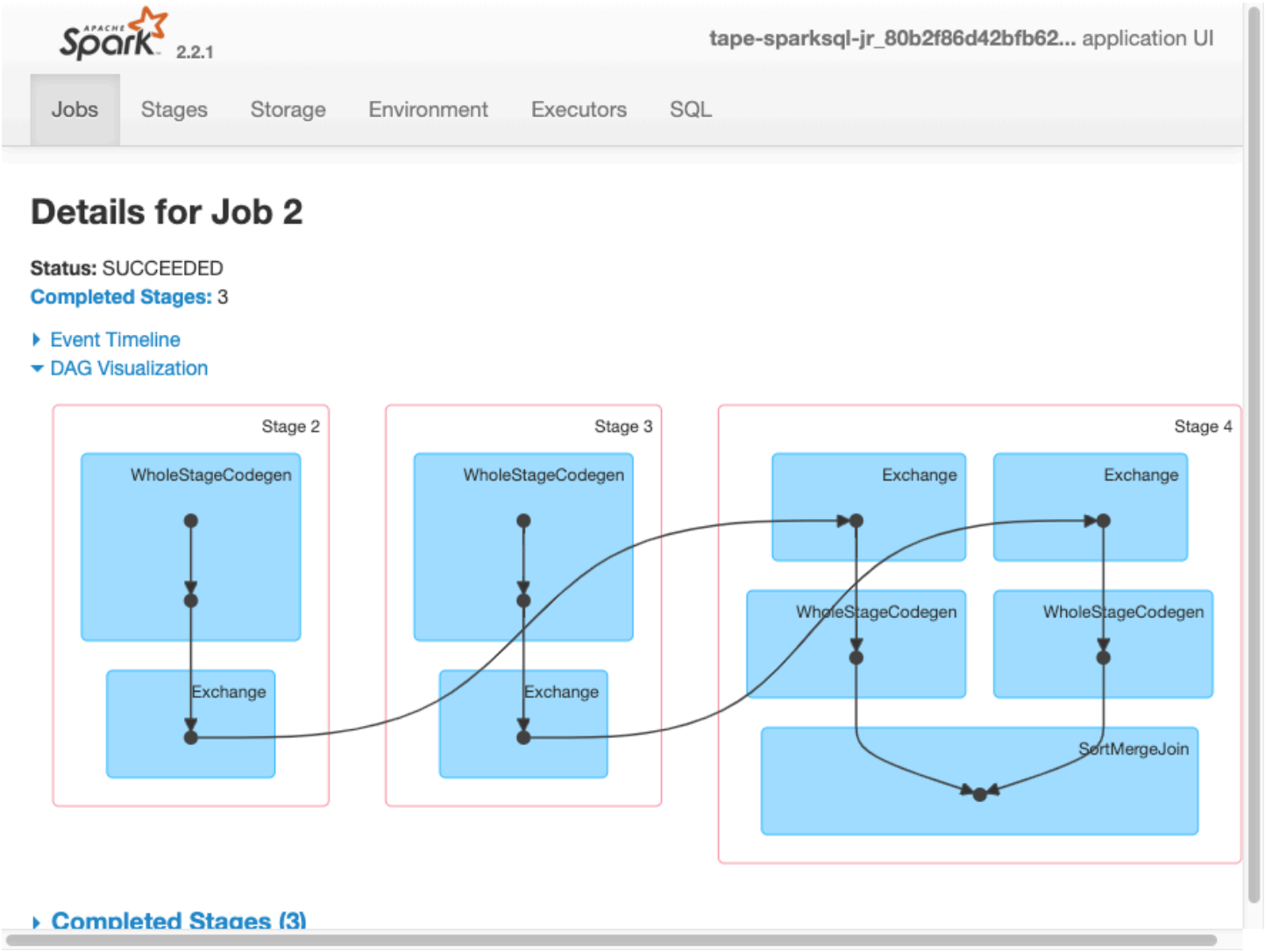
In diesem Beispiel wird veranschaulicht, wie Sie die Spark-Benutzeroberfläche verwenden, um Ihre Auftragsleistung nachzuvollziehen. Die Screenshots zeigen die Spark-Weboberfläche, wie sie von einem selbstverwalteten Spark History-Server bereitgestellt wird. Die Spark-Benutzeroberfläche in der AWS Glue Konsole bietet ähnliche Ansichten. Weitere Informationen zur Verwendung der Spark-Web-UI finden Sie unter Web-UI in der Spark-Dokumentation.

Im Folgenden finden Sie ein Beispiel für eine Spark-Anwendung, die aus zwei Datenquellen liest, eine Join-Transformation ausführt und diese im Parquet-Format zu Amazon S3 schreibt.

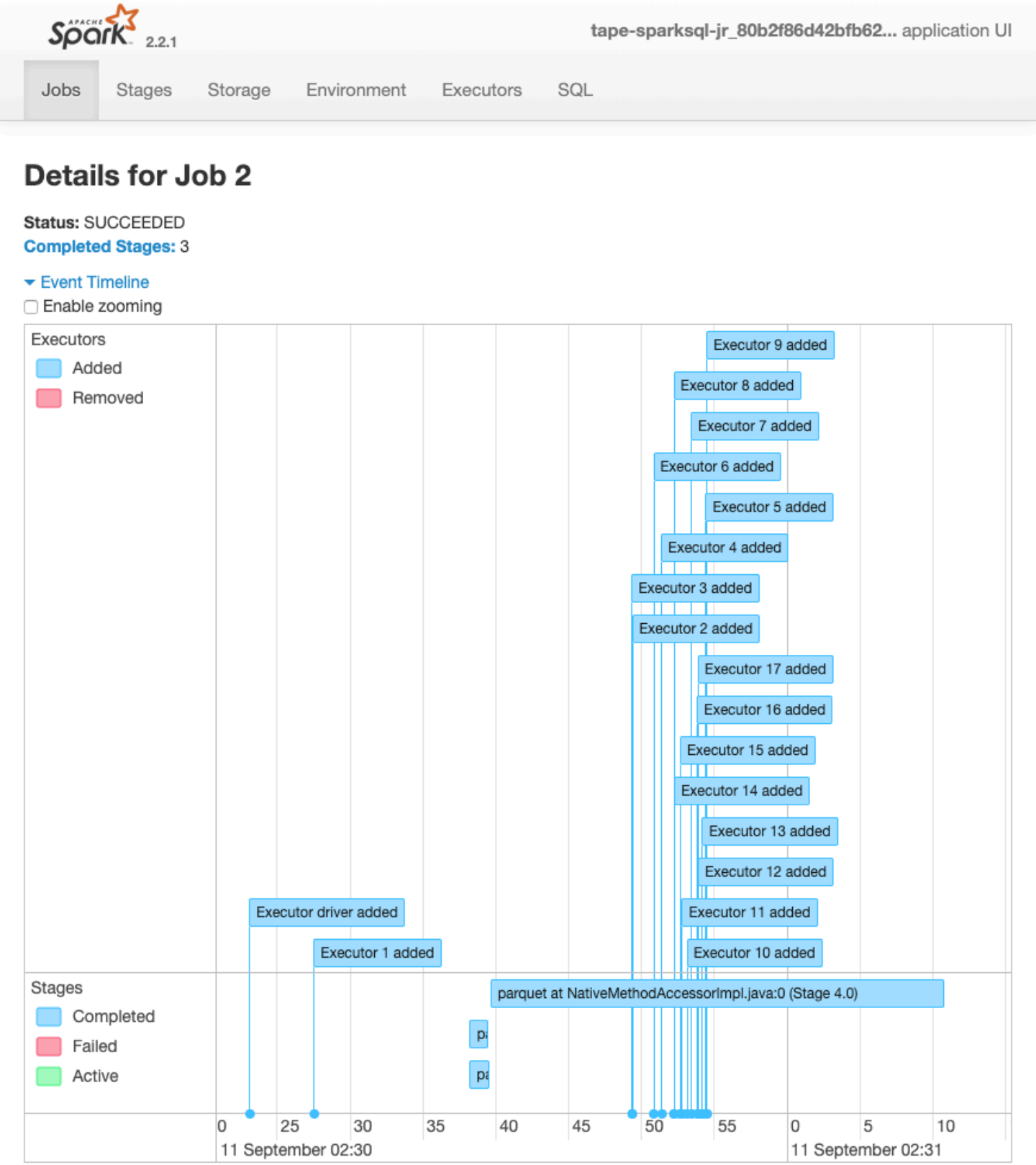
```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.functions import count, when, expr, col, sum, isnull
from pyspark.sql.functions import countDistinct
from awsglue.dynamicframe import DynamicFrame

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
```

```
job.init(args['JOB_NAME'])

df_persons = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
persons.json")
df_memberships = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
memberships.json")

df_joined = df_persons.join(df_memberships, df_persons.id == df_memberships.person_id,
 'fullouter')
df_joined.write.parquet("s3://aws-glue-demo-sparkui/output/")

job.commit()
```

Die folgende DAG-Visualisierung zeigt die verschiedenen Phasen in dieser Spark-Aufgabe.

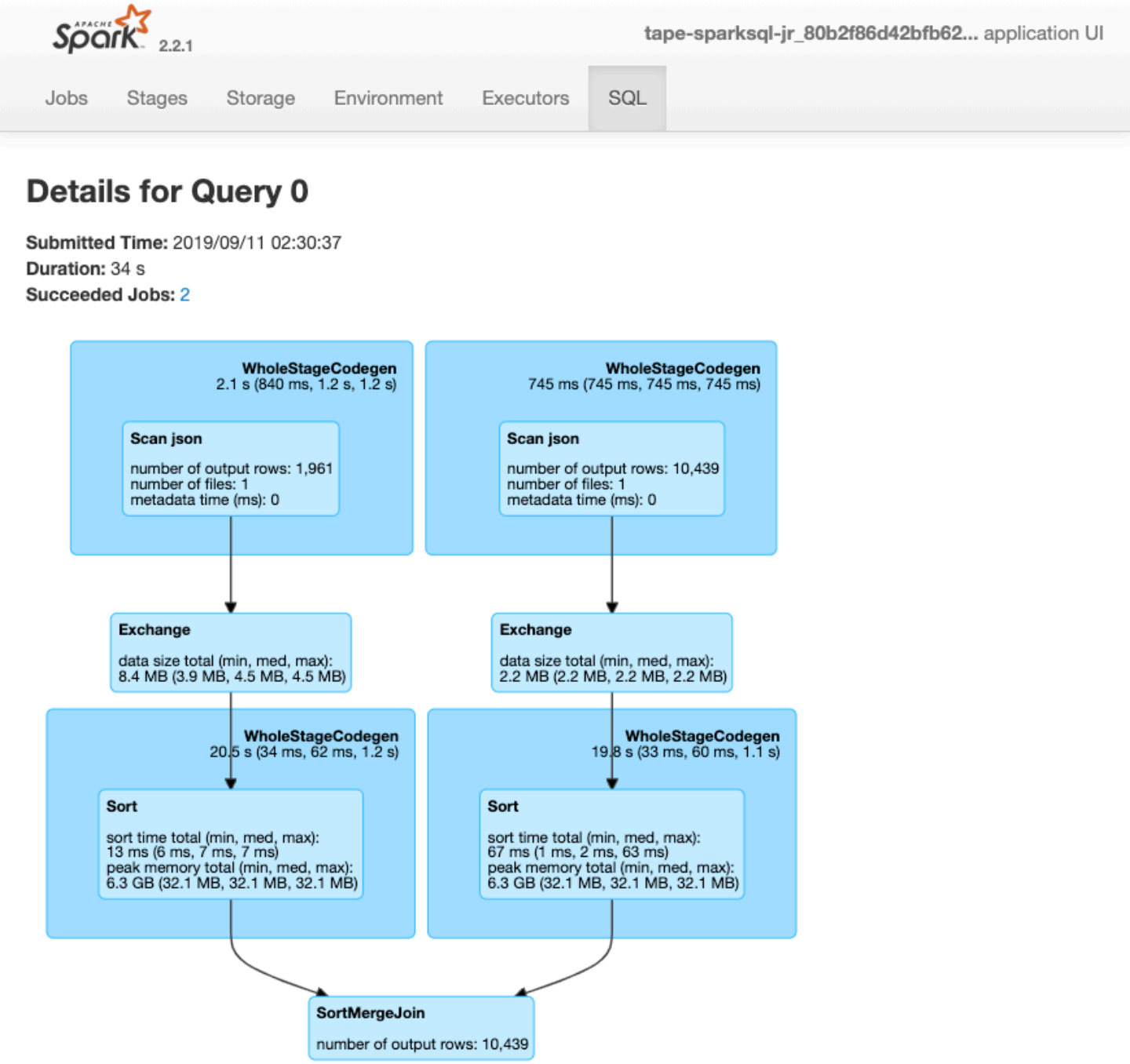


Der folgende Ereigniszeitplan für eine Aufgabe zeigt Start, Ausführung und Beendigung verschiedener Spark-Executors.


Spark 2.2.1
tape-sparksql-jr_80b2f86d42bfb62... application UI
Jobs
Stages
Storage
Environment
Executors
SQL
Details for Job 2
Status: SUCCEEDED
Completed Stages: 3
Event Timeline
Enable zooming
Executors
Added
Removed
Executor 9 added
Executor 8 added
Executor 7 added
Executor 6 added
Executor 5 added
Executor 4 added
Executor 3 added
Executor 2 added
Executor 17 added
Executor 16 added
Executor 15 added
Executor 14 added
Executor 13 added
Executor 12 added
Executor driver added
Executor 11 added
Executor 1 added
Executor 10 added
Stages
Completed
Failed
Active
parquet at NativeMethodAccessorImpl.java:0 (Stage 4.0)
0
25
30
35
40
45
50
55
0
5
10
11 September 02:30
11 September 02:31
DAG Visualization
Completed Stages (3)

Der folgende Bildschirm zeigt die Details der SparkSQL-Abfragepläne:

- Geparster logischer Plan
- Analysierter logischer Plan
- Optimierter logischer Plan
- Physischer Plan für die Ausführung


Spark 2.2.1
tape-sparksql-jr_80b2f86d42bfb62... application UI
Jobs
Stages
Storage
Environment
Executors
SQL
Details for Query 0
Submitted Time: 2019/09/11 02:30:37
Duration: 34 s
Succeeded Jobs: 2
WholeStageCodegen
2.1 s (840 ms, 1.2 s, 1.2 s)
Scan json
number of output rows: 1,961
number of files: 1
metadata time (ms): 0
WholeStageCodegen
745 ms (745 ms, 745 ms, 745 ms)
Scan json
number of output rows: 10,439
number of files: 1
metadata time (ms): 0
Exchange
data size total (min, med, max):
8.4 MB (3.9 MB, 4.5 MB, 4.5 MB)
Exchange
data size total (min, med, max):
2.2 MB (2.2 MB, 2.2 MB, 2.2 MB)
WholeStageCodegen
20.5 s (34 ms, 62 ms, 1.2 s)
Sort
sort time total (min, med, max):
13 ms (6 ms, 7 ms, 7 ms)
peak memory total (min, med, max):
6.3 GB (32.1 MB, 32.1 MB, 32.1 MB)
WholeStageCodegen
19.8 s (33 ms, 60 ms, 1.1 s)
Sort
sort time total (min, med, max):
67 ms (1 ms, 2 ms, 63 ms)
peak memory total (min, med, max):
6.3 GB (32.1 MB, 32.1 MB, 32.1 MB)
SortMergeJoin
number of output rows: 10,439
Details
== Parsed Logical Plan ==
Join FullOuter, (id#14 = person_id#50)
:-
Relation[birth_date#8,contact_details#9,death_date#10,family_name#11,gender#12,given_name#13,id#14,identifiers#15
,image#16,images#17,links#18,name#19,other_names#20,sort_name#21] json
+-
Relation[area_id#45,end_date#46,legislative_period_id#47,on_behalf_of_id#48,organization_id#49,person_id#50,role#
51,start_date#52] json
== Analyzed Logical Plan ==
birth_date: string, contact_details: array<struct<type:string,value:string>>, death_date: string, family_name:
string, gender: string, given_name: string, id: string, identifiers:
array<struct<identifier:string,scheme:string>>, image: string, images: array<struct<url:string>>, links:
array<struct<note:string,url:string>>, name: string, other_names:
array<struct<lang:string,name:string,note:string>>, sort_name: string, area_id: string, end_date: string,
legislative_period_id: string, on_behalf_of_id: string, organization_id: string, person_id: string, role: string,
start_date: string
Join FullOuter, (id#14 = person_id#50)

Themen

- Aktivierung der Apache Spark-Weboberfläche für AWS Glue jobs
- Starten des Spark History-Servers

Aktivierung der Apache Spark-Weboberfläche für AWS Glue jobs

Sie können die Apache Spark-Weboberfläche zum Überwachen und Debuggen verwenden AWS Glue ETL-Jobs laufen auf dem AWS Glue Jobsystem. Sie können die Spark-Benutzeroberfläche mit dem konfigurieren AWS Glue Konsole oder die AWS Command Line Interface (AWS CLI).

Alle 30 Sekunden AWS Glue sichert die Spark-Ereignisprotokolle auf dem von Ihnen angegebenen Amazon S3 S3-Pfad.

Themen

- Konfigurieren der Spark-Benutzeroberfläche (Konsole)
- Konfigurieren der Spark-Benutzeroberfläche (AWS CLI)
- Konfigurieren der Spark-Benutzeroberfläche für Sitzungen mit Notebooks
- Aktivieren Sie fortlaufende Logs

Konfigurieren der Spark-Benutzeroberfläche (Konsole)

Gehen Sie wie folgt vor, um die Spark-Benutzeroberfläche über die AWS Management Console zu konfigurieren. Beim Erstellen eines AWS Glue Jobs ist die Spark-Benutzeroberfläche standardmäßig aktiviert.

Die Spark-Benutzeroberfläche aktivieren, wenn Sie einen Auftrag erstellen oder bearbeiten

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
3. Wählen Auftrag hinzufügen oder einen vorhandenen Auftrag aus.
4. Klicken Sie in den Auftragsdetails auf Erweiterte Eigenschaften.
5. Wählen Sie auf der Registerkarte Spark-Benutzeroberfläche die Option Spark-UI-Protokolle in Amazon S3 schreiben aus.
6. Geben Sie einen Amazon-S3-Pfad zum Speichern der Spark-Ereignisprotokolle für die Aufgabe an. Beachten Sie, dass die Verschlüsselung auch auf die Protokolldatei der Spark-

Benutzeroberfläche angewendet wird, wenn Sie im Auftrag eine Sicherheitskonfiguration verwenden. Weitere Informationen finden Sie unter Verschlüsseln von Daten, die von AWS Glue geschrieben werden.

7. Gehen Sie unter Konfiguration der Protokollierung und Überwachung in der Spark-Benutzeroberfläche wie folgt vor:

   - Wählen Sie Standard, wenn Sie Protokolle zur Anzeige in der AWS Glue Konsole generieren.
   - Wenn Sie Protokolle zur Anzeige auf einem Spark History-Server generieren, wählen Sie Legacy aus.
   - Sie können auch beide Protokolltypen generieren.

Konfigurieren der Spark-Benutzeroberfläche (AWS CLI)

Um Protokolle für die Anzeige mit der Spark-Benutzeroberfläche zu generieren, verwenden Sie in der AWS Glue Konsole die, AWS CLI um die folgenden Jobparameter zu übergeben an AWS Glue Jobs. Weitere Informationen finden Sie unter the section called “Auftragsparameter”.

```
'--enable-spark-ui': 'true',
'--spark-event-logs-path': 's3://s3-event-log-path'
```

Um Protokolle an ihre Legacy-Speicherorte zu verteilen, setzen Sie den Parameter `--enable-spark-ui-legacy-path` auf `"true"`. Wenn Sie keine Protokolle in den beiden Formaten generieren möchten, entfernen Sie den Parameter `--enable-spark-ui`.

Konfigurieren der Spark-Benutzeroberfläche für Sitzungen mit Notebooks

> **Warning**
>
> AWS Glue interaktive Sitzungen unterstützen derzeit die Spark-Benutzeroberfläche in der Konsole nicht. Konfigurieren Sie einen Spark History-Server.

Wenn Sie AWS Glue Notebooks verwenden, richten Sie die SparkUI-Konfiguration ein, bevor Sie die Sitzung starten. Verwenden Sie dazu das `%%configure`-Zellen-Magic:

```
%%configure { “--enable-spark-ui”: “true”, “--spark-event-logs-path”: “s3://path” }
```

Aktivieren Sie fortlaufende Logs

Die Aktivierung von SparkUI- und Rolling-Log-Ereignisdateien für AWS Glue Jobs bietet mehrere Vorteile:

- Rolling Log Event Files — Wenn die rollierenden Log-Ereignisdateien aktiviert sind, werden separate Protokolldateien für jeden Schritt der Jobausführung AWS Glue generiert, sodass Probleme, die für eine bestimmte Phase oder Transformation spezifisch sind, leichter identifiziert und behoben werden können.
- Bessere Protokollverwaltung — Rollende Protokollereignisdateien helfen dabei, Protokolldateien effizienter zu verwalten. Anstatt über eine einzige, potenziell große Protokolldatei zu verfügen, werden die Protokolle je nach den Phasen der Auftragsausführung in kleinere, besser verwaltbare Dateien aufgeteilt. Dies kann die Archivierung, Analyse und Problembehandlung von Protokollen vereinfachen.
- Verbesserte Fehlertoleranz — Wenn ein AWS Glue Job fehlschlägt oder unterbrochen wird, können die fortlaufenden Protokollereignisdateien wertvolle Informationen über die letzte erfolgreiche Phase liefern, sodass es einfacher ist, den Job an diesem Punkt fortzusetzen, anstatt von vorne zu beginnen.
- Kostenoptimierung — Durch die Aktivierung von rollierenden Protokollereignisdateien können Sie die mit Protokolldateien verbundenen Speicherkosten sparen. Anstatt eine einzelne, potenziell große Protokolldatei zu speichern, speichern Sie kleinere, besser verwaltbare Protokolldateien, was insbesondere bei lang andauernden oder komplexen Aufträgen kostengünstiger sein kann.

In einer neuen Umgebung können Benutzer das Rolling Logs explizit aktivieren, indem sie:

```
'–conf': 'spark.eventLog.rolling.enabled=true'
```

or

```
'–conf': 'spark.eventLog.rolling.enabled=true –conf
spark.eventLog.rolling.maxFileSize=128m'
```

Wenn Rolling-Logs aktiviert sind, `spark.eventLog.rolling.maxFileSize` gibt dies die maximale Größe der Ereignisprotokolldatei an, bevor sie übertragen wird. Der Standardwert dieses optionalen Parameters ist 128 MB, sofern er nicht angegeben wird. Der Mindestwert ist 10 MB.

Die maximale Summe aller generierten gerollten Protokollereignisdateien beträgt 2 GB. Für AWS Glue Jobs ohne Rolling-Log-Unterstützung beträgt die maximale Größe der Protokollereignisdatei, die für SparkUI unterstützt wird, 0,5 GB.

Sie können fortlaufende Protokolle für einen Streaming-Auftrag deaktivieren, indem Sie eine zusätzliche Konfiguration übergeben. Beachten Sie, dass die Pflege sehr großer Protokolldateien kostspielig sein kann.

Um fortlaufende Protokolle zu deaktivieren, stellen Sie die folgende Konfiguration bereit:

```
'--spark-ui-event-logs-path': 'true',
'--conf': 'spark.eventLog.rolling.enabled=false'
```

Starten des Spark History-Servers

Mit einem Spark History-Server können Sie die Spark-Protokolle in Ihrer eigenen Infrastruktur visualisieren. Sie können dieselben Visualisierungen in der AWS Glue Konsole für AWS Glue Jobausführungen auf Versionen AWS Glue 4.0 oder höher sehen, wobei die Logs im Standardformat (und nicht im Legacy-Format) generiert wurden. Weitere Informationen finden Sie unter the section called “Überwachen über die Spark-Benutzeroberfläche”.

Sie können den Spark-Verlaufsserver mithilfe einer AWS CloudFormation Vorlage starten, die den Server auf einer EC2 Instance hostet, oder ihn lokal mit Docker starten.

Themen

- Den Spark-History-Server starten und die Spark-Benutzeroberfläche anzeigen mit AWS CloudFormation
- Starten des Spark History-Servers und Anzeigen der Spark-Benutzeroberfläche über Docker

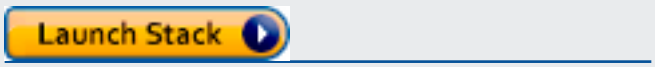
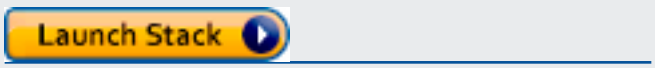
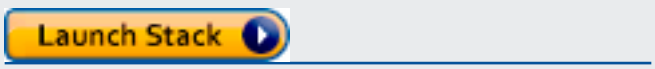
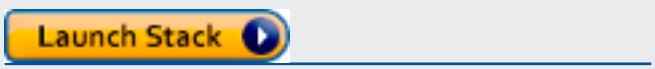
Den Spark-History-Server starten und die Spark-Benutzeroberfläche anzeigen mit AWS CloudFormation

Sie können eine AWS CloudFormation Vorlage verwenden, um den Apache Spark-History-Server zu starten und die Spark-Weboberfläche anzuzeigen. Diese Vorlagen sind Beispiele, die Sie entsprechend Ihren Anforderungen anpassen sollten.

Um den Spark-Geschichtsserver zu starten und die Spark-Benutzeroberfläche anzuzeigen, verwenden Sie AWS CloudFormation

1. Wählen Sie eine der Schaltflächen Launch Stack (Stack starten) in der folgenden Tabelle aus. Dadurch wird der Stack auf der AWS CloudFormation Konsole gestartet.

Region	Starten
US East (Ohio)	
USA Ost (Nord-Virginia)	
USA West (Nordkalifornien)	
USA West (Oregon)	
Africa (Cape Town)	
Asia Pacific (Hong Kong)	
Asia Pacific (Mumbai)	
Asien-Pazifik (Osaka)	
Asien-Pazifik (Seoul)	
Asien-Pazifik (Singapur)	
Asien-Pazifik (Sydney)	
Asien-Pazifik (Tokio)	

Region	Starten
Canada (Central)	
Europe (Frankfurt)	
Europa (Irland)	
Europa (London)	
Europa (Milan)	
Europe (Paris)	
Europe (Stockholm)	
Middle East (Bahrain)	
Südamerika (São Paulo)	

2. Wählen Sie auf der Seite Specify template (Vorlage angeben) die Option Next (Weiter) aus.
3. Geben Sie auf der Seite Specify stack details (Stack-Details angeben) den Wert für Stack name (Stack-Name) ein. Geben Sie zusätzliche Informationen unter Parameter ein.

   a. Konfiguration der Spark-Benutzeroberfläche

      Geben Sie die folgenden Informationen ein:

      - IP-Adressbereich – Der IP-Adressbereich, der zum Anzeigen der Spark-Benutzeroberfläche verwendet werden kann. Wenn Sie den Zugriff in Bezug auf einen bestimmten IP-Adressbereich einschränken möchten, sollten Sie einen benutzerdefinierten Wert verwenden.
      - History-Server-Port – Der Port für die Spark-Benutzeroberfläche. Sie können den Standardwert verwenden.

- Verzeichnis der Ereignisprotokolle — Wählen Sie den Speicherort aus, an dem Spark-Ereignisprotokolle gespeichert werden AWS Glue Job- oder Entwicklungsendpunkte. Für das Pfadschema der Ereignisprotokolle müssen Sie **`s3a://`** verwenden.
- Spark-Package-Standort – Sie können den Standardwert verwenden.
- KeyStore-Pfad – SSL/TLS-KeyStore-Pfad für HTTPS. Wenn Sie eine benutzerdefinierte Schlüsselspeicherdatei verwenden möchten, können Sie den S3-Pfad `s3://path_to_your_keystore_file` hier angeben. Wenn Sie diesen Parameter leer lassen, wird ein selbstsignierter, zertifikatbasierter Schlüsselspeicher generiert und verwendet.
- KeyStore-Passwort – Geben Sie SSL/TLS-KeyStore-Passwort für HTTPS ein.

b. EC2 Konfiguration der Instanz

Geben Sie die folgenden Informationen ein:

- Instance-Typ — Der Typ der EC2 Amazon-Instance, die den Spark-History-Server hostet. Da mit dieser Vorlage eine EC2 Amazon-Instance in Ihrem Konto gestartet wird, werden die EC2 Amazon-Kosten Ihrem Konto separat belastet.
- Neueste AMI-ID – Die AMI-ID von Amazon Linux 2 für die Spark-Verlaufsserver-Instance. Sie können den Standardwert verwenden.
- VPC ID – Die Virtual Private Cloud (VPC)-ID für die Spark-History-Server-Instance. Sie können alle in Ihrem Konto VPCs verfügbaren Optionen verwenden. Die Verwendung einer Standard-VPC mit einer Standard-Netzwerk-ACL wird nicht empfohlen. Weitere Informationen finden Sie unter Default VPC and Default Subnets (Standard-VPC und Standard-Subnetze) und unter Creating a VPC (Erstellen einer VPC) im Amazon-VPC-Benutzerhandbuch.
- Subnetz-ID – Die ID für die Spark-Verlaufsserver-Instance. Sie können jedes Subnetz in Ihrer VPC verwenden. Sie müssen das Subnetz von Ihrem Client aus erreichen können. Wenn Sie über das Internet auf das Subnetz zugreifen möchten, müssen Sie ein öffentliches Subnetz mit dem Internet-Gateway in der Routing-Tabelle verwenden.

c. Wählen Sie Weiter.

4. Wählen Sie auf der Seite Stack-Optionen konfigurieren die Option Weiter aus, um anhand der aktuellen Benutzeranmeldedaten zu bestimmen, wie Ressourcen im Stack erstellt, geändert oder gelöscht werden CloudFormation können. Sie können im Abschnitt Berechtigungen auch eine Rolle angeben, die anstelle der aktuellen Benutzerberechtigungen verwendet werden soll, und dann Weiter wählen.

5. Überprüfen Sie die Vorlage auf der Seite Review (Prüfen).

   Wählen Sie Ich bestätige, dass AWS CloudFormation möglicherweise IAM-Ressourcen erstellt werden, und wählen Sie dann Stack erstellen aus.
6. Warten Sie, bis der Stack erstellt wurde.
7. Öffnen Sie die Registerkarte Outputs (Ausgaben).

   a. Kopieren Sie die URL von SparkUiPublicUrl, wenn Sie ein öffentliches Subnetz verwenden.

   b. Kopieren Sie die URL von SparkUiPrivateUrl, wenn Sie ein privates Subnetz verwenden.
8. Öffnen Sie einen Webbrowser und fügen Sie die URL ein. Auf diese Weise können Sie über HTTPS auf dem angegebenen Port auf den Server zugreifen. Ihr Browser erkennt möglicherweise nicht das Zertifikat des Servers. In diesem Fall müssen Sie den Schutz des Servers außer Kraft setzen und trotzdem fortfahren.

Starten des Spark History-Servers und Anzeigen der Spark-Benutzeroberfläche über Docker

Wenn Sie lokalen Zugriff bevorzugen (ohne eine EC2 Instanz für den Apache Spark-Geschichtsserver zu haben), können Sie auch Docker verwenden, um den Apache Spark-Geschichtsserver zu starten und die Spark-Benutzeroberfläche lokal anzuzeigen. Diese Dockerfile ist ein Beispiel, das Sie entsprechend Ihren Anforderungen anpassen sollten.

Voraussetzungen

Informationen zum Installieren von Docker auf Ihrem Laptop finden Sie in der Docker Engine-Community.

So starten Sie den Spark History-Server und zeigen die Spark-Benutzeroberfläche lokal über Docker an

1. Laden Sie Dateien herunter von GitHub.

   Laden Sie das Dockerfile herunter und von `pom.xml` AWS Glue Codebeispiele.
2. Bestimmen Sie, ob Sie Ihre Benutzeranmeldeinformationen oder Verbundbenutzeranmeldeinformationen für den Zugriff auf AWS verwenden möchten.

   - Um die aktuellen Benutzeranmeldedaten für den Zugriff zu verwenden AWS, rufen Sie die Werte ab, die für `AWS_ACCESS_KEY_ID` und `AWS_SECRET_ACCESS_KEY` im `docker run` Befehl verwendet werden sollen. Weitere Informationen finden Sie unter Verwalten der Zugriffsschlüssel für IAM-Benutzer im -IAM-Benutzerhandbuch.

- Um SAML 2.0-Verbundbenutzer für den Zugriff zu verwenden AWS, rufen Sie die Werte für `AWS_ACCESS_KEY_IDAWS_SECRET_ACCESS_KEY`, und ab. `AWS_SESSION_TOKEN` Weitere Informationen finden Sie unter Anfordern von temporären Sicherheitsanmeldeinformationen

3. Bestimmen Sie den Speicherort Ihres Ereignisprotokollverzeichnisses für die Verwendung im `docker run`-Befehl.
4. Erstellen Sie das Docker-Image mit den Dateien im lokalen Verzeichnis unter Verwendung des Namens `glue/sparkui`, und das Tag`latest`aus.

```
$ docker build -t glue/sparkui:latest .
```

5. Erstellen und starten Sie den Docker-Container.

   Verwenden Sie in den folgenden Befehlen die zuvor in den Schritten 2 und 3 beschafften Werte.

   a. Verwenden Sie einen Befehl wie den folgenden, um den Docker-Container mit Ihren Benutzeranmeldeinformationen zu erstellen

```
docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
Dspark.history.fs.logDirectory=s3a://path_to_eventlog
 -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY"
 -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
 org.apache.spark.deploy.history.HistoryServer"
```

   b. Um den Docker-Container mit temporären Anmeldeinformationen zu erstellen, verwenden Sie `org.apache.hadoop.fs.s3a.TemporaryAWSCredentialsProvider` als Anbieter und geben Sie die in Schritt 2 beschafften Anmeldeinformationen an. Weitere Informationen finden Sie unter Using Session Credentials with Temporary AWSCredentials Provider in der Dokumentation Hadoop: Integration with Amazon Web Services.

```
docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
Dspark.history.fs.logDirectory=s3a://path_to_eventlog
 -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY
 -Dspark.hadoop.fs.s3a.session.token=AWS_SESSION_TOKEN
 -
Dspark.hadoop.fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.TemporaryAWSCred
 -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
 org.apache.spark.deploy.history.HistoryServer"
```

> **Note**
>
> Diese Konfigurationsparameter stammen aus dem Hadoop-AWS Modul. Sie müssen möglicherweise je nach Anwendungsfall eine spezielle Konfiguration vornehmen. Zum Beispiel: Benutzer in isolierten Regionen müssen den `spark.hadoop.fs.s3a.endpoint` konfigurieren.

6. Öffnen Sie `http://localhost:18080` in Ihrem Browser, um die Spark-Benutzeroberfläche lokal anzuzeigen.

Überwachung mit Einblicken in die AWS Glue Auftragsausführung

AWS Glue Job Run Insights ist eine Funktion AWS Glue , die das Debuggen und die Optimierung von Jobs für Ihre AWS Glue Jobs vereinfacht. AWS Glue bietet die Spark-Benutzeroberfläche sowie CloudWatch Protokolle und Metriken für die Überwachung Ihrer AWS Glue Jobs. Mit dieser Funktion erhalten Sie folgende Informationen über die Ausführung Ihres AWS Glue Jobs:

- Zeilennummer Ihres AWS Glue Job-Skripts, bei dem ein Fehler aufgetreten ist.
- Spark-Aktion, die zuletzt im Spark-Abfrageplan kurz vor dem Fehler Ihres Auftrags ausgeführt wurde.
- Spark-Ausnahmeereignisse im Zusammenhang mit dem Fehler, der in einem zeitgeordneten Protokollstream dargestellt wird.
- Ursachenanalyse und empfohlene Maßnahmen (z. B. die Optimierung Ihres Skripts), um das Problem zu beheben.
- Häufige Spark-Ereignisse (Protokollmeldungen in Bezug auf eine Spark-Aktion) mit einer empfohlenen Aktion, die die Ursache behandelt.

All diese Erkenntnisse stehen Ihnen mithilfe von zwei neuen Log-Streams in den CloudWatch Logs für Ihre AWS Glue Jobs zur Verfügung.

Voraussetzungen

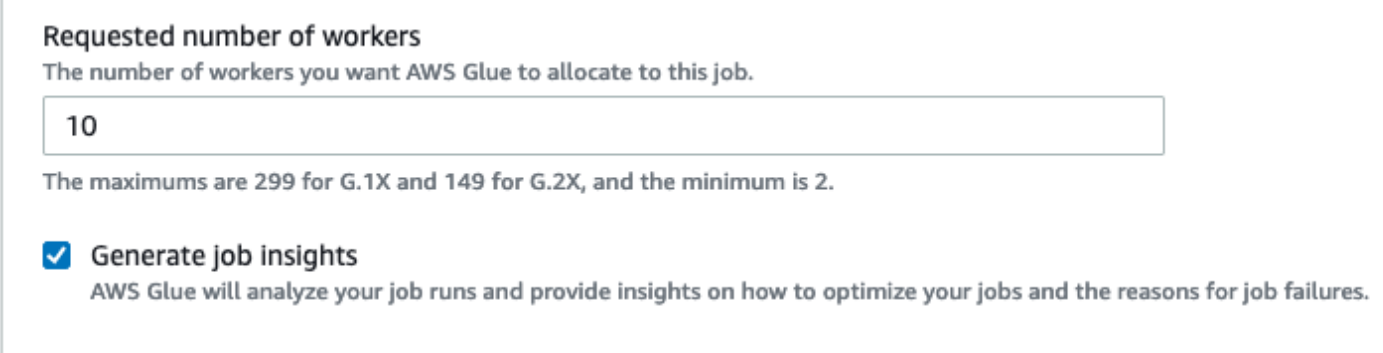
Die Funktion „Einblicke in die AWS Glue Auftragsausführung“ ist für die AWS Glue Versionen 2.0, 3.0, 4.0 und 5.0 verfügbar. Sie können dem Migrationsleitfaden für Ihre bestehenden Jobs folgen, um sie von älteren AWS Glue Versionen zu aktualisieren.

Einblicke in die Jobausführung für einen AWS Glue ETL-Job aktivieren

Sie können Einblicke in die Jobausführung über AWS Glue Studio oder über die CLI aktivieren.

AWS Glue Studio

Wenn Sie einen Job über erstellen AWS Glue Studio, können Sie die Einblicke in die Jobausführung auf der Registerkarte Jobdetails aktivieren oder deaktivieren. Vergewissern Sie sich, dass das Feld Job-Insights generieren ausgewählt ist.



Befehlszeile

Wenn Sie einen Auftrag über die CLI erstellen, können Sie eine Auftragsausführung mit einem einzigen neuen Auftrags-Parameter starten: `--enable-job-insights = true`.

Standardmäßig werden die Protokoll-Streams der Auftragsausführungs-Erkenntnisse unter derselben Standardprotokollgruppe erstellt, die von Kontinuierliche AWS Glue -Protokollierung, das heißt, `/aws-glue/jobs/logs-v2/` verwendet wird. Sie können benutzerdefinierte Protokollgruppennamen, Protokollfilter und Protokollgruppenkonfigurationen mit denselben Argumenten für die kontinuierliche Protokollierung einrichten. Weitere Informationen finden Sie unter Kontinuierliche Protokollierung für AWS Glue Jobs aktivieren.

Beim Zugriff auf den Job Run Insights werden Streams protokolliert in CloudWatch

Wenn das Feature „Auftragsausführungs-Erkenntnisse“ aktiviert ist, werden möglicherweise zwei Protokoll-Streams erstellt, wenn eine Auftragsausführung fehlschlägt. Wenn ein Auftrag erfolgreich abgeschlossen ist, wird keiner der Streams generiert.

1. Protokoll-Stream für Ausnahmeanalyse: `<job-run-id>-job-insights-rca-driver`. Dieser Stream bietet Folgendes:
   - Zeilennummer Ihres AWS Glue Job-Skripts, das den Fehler verursacht hat.

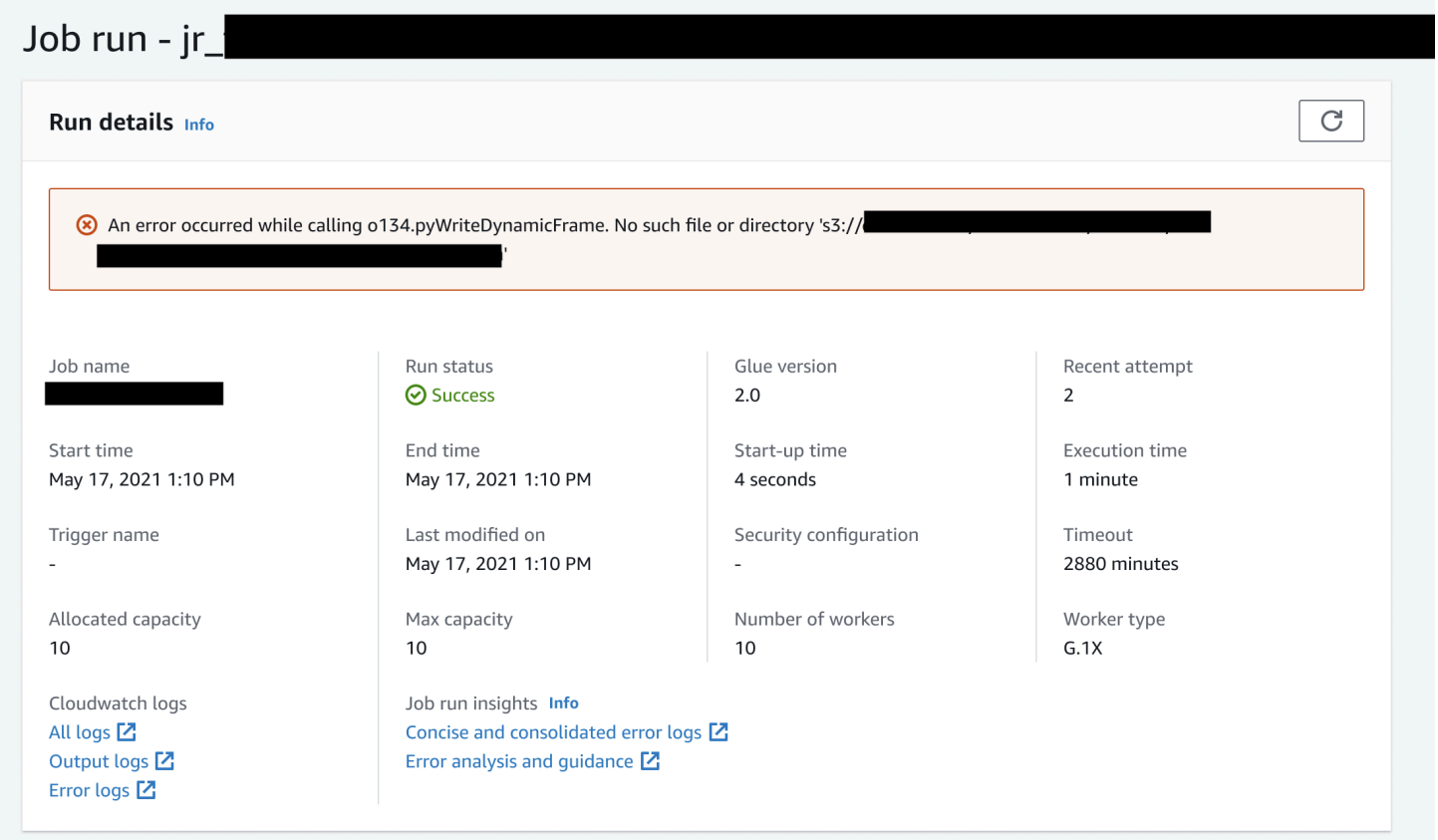
- Spark-Aktion, die zuletzt im Spark-Abfrageplan (DAG) ausgeführt wurde.
- Prägnante zeitgeordnete Ereignisse des Spark-Treibers und der Executors, die mit der Ausnahme zusammenhängen. Hier finden Sie Details wie vollständige Fehlermeldungen, die fehlgeschlagene Spark-Aufgabe und deren Executor-ID, die Ihnen helfen, sich bei Bedarf auf den Protokoll-Stream des jeweiligen Executors zu konzentrieren.

2. Regelbasierter Erkenntnis-Stream:

- Ursachenanalyse und Empfehlungen zur Behebung der Fehler (z. B. Verwendung eines bestimmten Auftrags-Parameters zur Optimierung der Leistung).
- Relevante Spark-Ereignisse, die als Grundlage für die Ursachenanalyse und eine empfohlene Aktion dienen.

> **Note**
>
> Der erste Stream wird nur vorhanden sein, wenn Spark-Ausnahme-Ereignisse für eine fehlgeschlagene Auftragsausführung verfügbar sind und der zweite Stream wird nur vorhanden sein, wenn Erkenntnisse für die fehlgeschlagene Auftragsausführung verfügbar sind. Wenn Ihr Auftrag beispielsweise erfolgreich abgeschlossen ist, wird keiner der Streams generiert. Wenn Ihr Auftrag fehlschlägt, aber keine dienstdefinierte Regel vorhanden ist, die mit Ihrem Fehlerszenario übereinstimmen kann, wird nur der erste Stream generiert.

Wenn der Job aus erstellt wurde AWS Glue Studio, sind die Links zu den oben genannten Streams auch auf der Registerkarte Job-Rundetails (Job Run Insights) als „Präzise und konsolidierte Fehlerprotokolle“ und „Fehleranalyse und Anleitung“ verfügbar.



Beispiel für Einblicke in die AWS Glue Auftragsausführung

In diesem Abschnitt stellen wir Ihnen ein Beispiel vor, wie das Feature „Auftragsausführungs-Erkenntnisse“ Ihnen helfen kann, ein Problem in Ihrem fehlgeschlagenen Auftrag zu lösen. In diesem Beispiel hat ein Benutzer vergessen, das erforderliche Modul (Tensorflow) in einen AWS Glue Job zu importieren, um seine Daten zu analysieren und ein Modell für maschinelles Lernen zu erstellen.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.types import *
from pyspark.sql.functions import udf,col

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
```

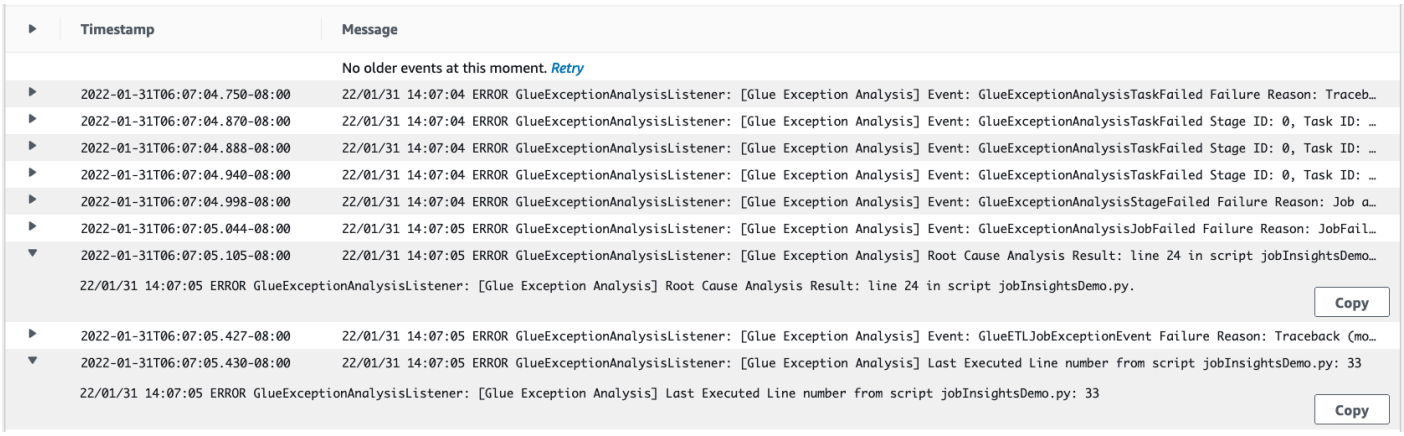
```
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

data_set_1 = [1, 2, 3, 4]
data_set_2 = [5, 6, 7, 8]

scoresDf = spark.createDataFrame(data_set_1, IntegerType())

def data_multiplier_func(factor, data_vector):
    import tensorflow as tf
    with tf.compat.v1.Session() as sess:
        x1 = tf.constant(factor)
        x2 = tf.constant(data_vector)
        result = tf.multiply(x1, x2)
        return sess.run(result).tolist()

data_multiplier_udf = udf(lambda x:data_multiplier_func(x, data_set_2),
 ArrayType(IntegerType(),False))
factoredDf = scoresDf.withColumn("final_value", data_multiplier_udf(col("value")))
print(factoredDf.collect())
```

Ohne das Feature „Auftragsausführungs-Erkenntnisse“ wird nur die folgende Nachricht angezeigt, die von Spark ausgelöst wird, da der Auftrag fehlschlägt:

```
An error occurred while calling o111.collectToPython. Traceback (most
recent call last):
```

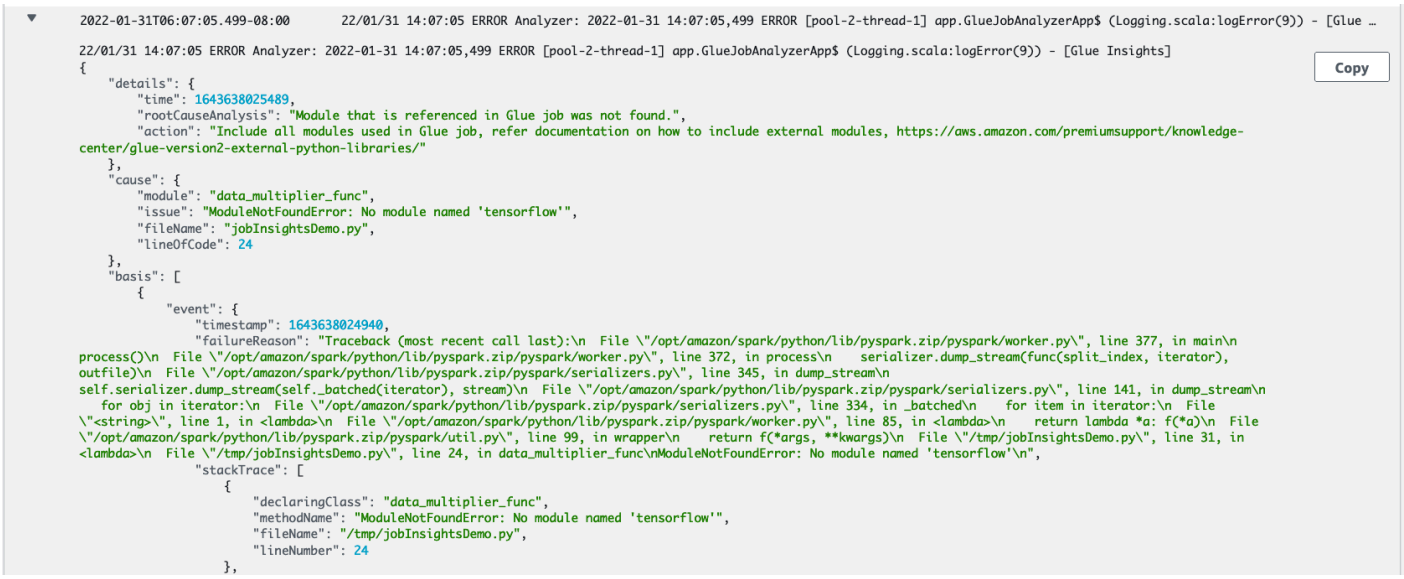
Die Nachricht ist mehrdeutig und schränkt Ihre Debugging-Erfahrung ein. In diesem Fall bietet Ihnen diese Funktion zusätzliche Einblicke in zwei CloudWatch Protokollstreams:

1. Der `job-insights-rca-driver`-Protokoll-Stream:
   - Ausnahme-Ereignisse: Dieser Protokoll-Stream bietet Ihnen die Spark-Ausnahmeereignisse im Zusammenhang mit dem Fehler, der vom Spark-Treiber und verschiedenen verteilten Workern gesammelt wurde. Diese Ereignisse helfen Ihnen, die zeitlich geordnete Ausbreitung der Ausnahme zu verstehen, wenn fehlerhafter Code auf Spark-Aufgaben, Executoren und Stages, die auf die Worker verteilt sind, ausgeführt wird. AWS Glue
   - Zeilennummern: Dieser Protokoll-Stream identifiziert Zeile 21, die den Aufruf zum Importieren des fehlenden Python-Moduls gemacht hat, das den Fehler verursacht hat; Außerdem wird Zeile 24, der Aufruf von Spark Action `collect()`, als die zuletzt ausgeführte Zeile in Ihrem Skript identifiziert.



2. Der `job-insights-rule-driver`-Protokoll-Stream:

   - Grundursache und Empfehlung: Neben der Zeilennummer und der zuletzt ausgeführten Zeilennummer für den Fehler in Ihrem Skript zeigt dieser Protokollstream die Ursachenanalyse und die Empfehlung für Sie, dem AWS Glue Dokument zu folgen und die erforderlichen Jobparameter einzurichten, um ein zusätzliches Python-Modul in Ihrem AWS Glue Job zu verwenden.
   - Basis-Ereignis: Dieser Protokoll-Stream zeigt auch das Spark-Ausnahmeereignis an, das mit der vom Service definierten Regel ausgewertet wurde, um die Ursache abzuleiten und eine Empfehlung abzugeben.



Überwachung mit Amazon CloudWatch

Sie können überwachen AWS Glue mithilfe von Amazon CloudWatch, das Rohdaten sammelt und verarbeitet von AWS Glue in lesbare near-real-time Metriken. Diese Statistiken werden für

einen Zeitraum von zwei Wochen aufgezeichnet, damit Sie auf Verlaufsinformationen zugreifen können und einen besseren Überblick über die Performance der Webanwendung oder des Services erhalten. Standardmäßig AWS Glue Metrikdaten werden CloudWatch automatisch gesendet. Weitere Informationen finden Sie unter Was ist Amazon CloudWatch? im CloudWatch Amazon-Benutzerhandbuch undAWS Glue Metriken.

Kontinuierliche Protokollierung

AWS Glue unterstützt auch die kontinuierliche Protokollierung in Echtzeit für AWS Glue Jobs. Wenn die kontinuierliche Protokollierung für einen Job aktiviert ist, können Sie die Echtzeitprotokolle auf der AWS Glue Konsole oder das CloudWatch Konsolen-Dashboard. Weitere Informationen finden Sie unter Protokollierung für AWS Glue Jobs.

Beobachtbarkeitsmetriken

Wenn Job-Observability-Metriken aktiviert sind, werden zusätzliche Amazon CloudWatch Metriken generiert, wenn der Job ausgeführt wird. Verwenden Sie AWS Glue Messwerte zur Beobachtbarkeit, um Einblicke in das zu gewinnen, was in Ihrem System vor sich geht AWS Glue um die Triaging und Analyse von Problemen zu verbessern.

Themen

- Überwachen AWS Glue Verwendung von CloudWatch Amazon-Metriken
- CloudWatch Amazon-Alarme einrichten AWS Glue Berufsprofile
- Protokollierung für AWS Glue Jobs
- Überwachen mit AWS Glue Beobachtbarkeitsmetriken

Überwachen AWS Glue Verwendung von CloudWatch Amazon-Metriken

Sie können Profile erstellen und überwachen AWS Glue Operationen mit AWS Glue Job-Profiler. Es sammelt und verarbeitet Rohdaten von AWS Glue Jobs in lesbare, nahezu in Echtzeit gespeicherte Metriken, die in Amazon gespeichert sind CloudWatch. Diese Statistiken werden gespeichert und aggregiert, CloudWatch sodass Sie auf historische Informationen zugreifen können, um einen besseren Überblick über die Leistung Ihrer Anwendung zu erhalten.

> **Note**
>
> Es können zusätzliche Gebühren anfallen, wenn Sie Job-Metriken aktivieren und CloudWatch benutzerdefinierte Metriken erstellt werden. Weitere Informationen finden Sie unter CloudWatch Amazon-Preise.

AWS Glue Überblick über die Kennzahlen

Wenn Sie interagieren mit AWS Glue, es sendet Metriken an CloudWatch. Sie können diese Metriken mit dem AWS Glue console (die bevorzugte Methode), das CloudWatch Konsolen-Dashboard oder die AWS Command Line Interface (AWS CLI).

Um Metriken mit dem anzuzeigen AWS Glue Konsolen-Dashboard

Sie können zusammenfassende oder detaillierte Diagramme von Metriken für einen Auftrag oder detaillierte Diagramme für eine Auftragsausführung anzeigen.

1. Melden Sie sich bei an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Überwachung der Auftragsausführung aus.
3. Wählen Sie unter Auftragsausführungen die Option Aktionen, um einen gerade ausgeführten Auftrag anzuhalten, einen Auftrag anzuzeigen oder ein Auftragslesezeichen zurückzuspulen.
4. Wählen Sie einen Auftrag aus und wählen Sie dann Details zur Ausführung anzeigen, um zusätzliche Informationen zur Auftragsausführung anzuzeigen.

Um Metriken über das CloudWatch Konsolen-Dashboard anzuzeigen

Metriken werden zunächst nach dem Service-Namespace und anschließend nach den verschiedenen Dimensionskombinationen in den einzelnen Namespaces gruppiert.

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie den Namespace Glue aus.

Um Metriken mit dem anzuzeigen AWS CLI

- Geben Sie als Eingabeaufforderung den folgenden Befehl ein.

```
aws cloudwatch list-metrics --namespace Glue
```

AWS Glue meldet Metriken CloudWatch alle 30 Sekunden, und die CloudWatch Metrik-Dashboards sind so konfiguriert, dass sie jede Minute angezeigt werden. Das Tool AWS Glue Metriken stellen Deltawerte aus den zuvor gemeldeten Werten dar. Gegebenenfalls aggregieren (summieren) die Metrik-Dashboards die 30-Sekunden-Wertezu einem Wert für die gesamte vergangene Minute.

AWS Glue Metriken, Verhalten für Spark-Jobs

AWS Glue Metriken werden bei der Initialisierung von `GlueContext` in einem Skript aktiviert und in der Regel erst am Ende einer Apache Spark-Aufgabe aktualisiert. Sie stellen die aggregierten Werte in allen abgeschlossenen Spark-Aufgaben dar.

Allerdings sind die Spark-Metriken AWS Glue CloudWatch Es handelt sich im Allgemeinen um absolute Werte, die den aktuellen Status zum Zeitpunkt der Meldung angeben. AWS Glue meldet sie CloudWatch alle 30 Sekunden, und die Metrik-Dashboards zeigen in der Regel den Durchschnitt der in der letzten Minute empfangenen Datenpunkte an.

AWS Glue Vor den Namen der Metriken steht jeweils einer der folgenden Präfixtypen:

- `glue.driver.`— Metriken, deren Namen mit diesem Präfix beginnen, stehen entweder AWS Glue Metriken, die von allen Executoren auf dem Spark-Treiber aggregiert wurden, oder Spark-Metriken, die dem Spark-Treiber entsprechen.
- `glue.executorId.` – Die executorId ist die Nummer eines bestimmten Spark-Executor. Der Wert entspricht den Executors, die in den Protokollen aufgeführt sind.
- `glue.ALL.` - Metriken, deren Namen mit diesem Präfix beginnen, aggregierte Werte aus allen Spark Executors.

AWS Glue Metriken

AWS Glue erstellt und sendet CloudWatch alle 30 Sekunden die folgenden Metriken, und das AWS Glue Metrics-Dashboard meldet sie einmal pro Minute:

Metrik	Beschreibung
`glue.driver.aggregate.bytesRead`	Die Anzahl der Bytes, die von allen abgeschlossenen Spark-Aufgaben von allen Datenquellen gelesen und in allen Executors ausgeführt werden. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Byte Kann für die Überwachung folgender Aspekte verwendet werden: • Gelesene Bytes. • Fortschritt des Auftrags. • JDBC-Datenquellen. • Probleme mit Lesezeichen. • Abweichung über Auftragsausführungen hinweg. Diese Metrik kann genauso verwendet werden wie die `glue.ALL.s3.filesystem.read_bytes` -Metrik mit dem Unterschied, dass diese Metrik am Ende einer Spark-Aufgabe aktualisiert wird und auch Nicht-S3-Datenquellen erfasst.
`glue.driver.aggregate.elapsedTime`	Die ETL verstrichene Zeit in Millisekunden (schließt die Bootstrap-Zeiten des Auftrags nicht ein).

Metrik	Beschreibung
	Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Millisekunden Kann verwendet werden, um zu ermitteln, wie lange eine Auftragsausführung im Durchschnitt dauert. Einige Möglichkeiten, die Daten zu verwenden: • Stellen Sie Alarme für Straggler ein. • Messen Sie die Abweichung über Auftragsa usführungen hinweg.

Metrik	Beschreibung
`glue.driver.aggregate.numCompletedStages`	Die Anzahl der abgeschlossenen Phasen im Auftrag. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Fortschritt des Auftrags. • Zeitachse pro Stufe der Auftragsausführung, wenn mit anderen Metriken korreliert. Einige Möglichkeiten, die Daten zu verwenden: • Identifizieren Sie anspruchsvolle Phasen bei der Ausführung eines Auftrags. • Stellen Sie Alarme für korrelierte Spikes (anspruchsvolle Phasen) über Auftragsausführungen hinweg ein.

Metrik	Beschreibung
`glue.driver.aggregate.numCompletedTasks`	Die Anzahl der abgeschlossenen Aufgaben im Auftrag. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Fortschritt des Auftrags. • Parallelität innerhalb einer Phase.

Metrik	Beschreibung
`glue.driver.aggregate.numFailedTasks`	Die Anzahl der fehlgeschlagenen Aufgaben. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Datenanomalien, die zum Scheitern von Aufträgen führen. • Clusteranomalien, die zum Scheitern von Aufträgen führen. • Skriptanomalien, die zum Scheitern von Aufträgen führen. Die Daten können verwendet werden, um Alarme für erhöhte Ausfälle einzustellen, die Anomalien in Daten, Clustern oder Skripten andeuten könnten.

Metrik	Beschreibung
`glue.driver.aggregate.numKilledTasks`	Anzahl der abgeschlossenen Aufgaben. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Unregelmäßigkeiten bei der Datenverzerrung, die zu Ausnahmen (OOMs) führen, die Aufgaben zum Erliegen bringen. • Abnormalitäten im Skript, die zu Ausnahmen (OOMs) führen, die Aufgaben beenden. Einige Möglichkeiten, die Daten zu verwenden: • Stellen Sie Alarme für erhöhte Ausfälle ein, die Datenanomalien anzeigen. • Stellen Sie Alarme für erhöhte Ausfälle ein, die Clusteranomalien anzeigen. • Stellen Sie Alarme für erhöhte Ausfälle ein, die Skriptanomalien anzeigen.

Metrik	Beschreibung
`glue.driver.aggregate.recordsRead`	Die Anzahl der Datensätze, die von allen abgeschlossenen Spark-Aufgaben von allen Datenquellen gelesen und in allen Executors ausgeführt werden. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Gelesene Datensätze. • Fortschritt des Auftrags. • JDBC-Datenquellen. • Probleme mit Lesezeichen. • Verzerrung in Auftragsausführungen über mehrere Tage. Diese Metrik kann genauso verwendet werden wie die `glue.ALL.s3.filesystem.read_bytes` -Metrik mit dem Unterschied, dass diese Metrik am Ende einer Spark-Aufgabe aktualisiert wird.

Metrik	Beschreibung
`glue.driver.aggregate.shuffleBytesWritten`	Die Anzahl der Byte, die seit dem letzten Bericht von allen Executoren geschrieben wurden, um Daten zwischen ihnen zu mischen (aggregiert vom AWS Glue Metrics Dashboard als die Anzahl der zu diesem Zweck in der letzten Minute geschriebenen Byte). Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Byte Kann verwendet werden, um Folgendes zu überwachen: Datenmischung in Aufträgen (große Joins, GroupBy, Repartition, Coalesce). Einige Möglichkeiten, die Daten zu verwenden: • Repartitionieren oder Dekomprimieren Sie große Eingabedateien vor der weiteren Verarbeitung. • Repartitionieren Sie Daten einheitlicher, um Hotkeys zu vermeiden. • Vorfiltern von Daten vor Joins oder GroupBy-Operationen.

Metrik	Beschreibung
`glue.driver.aggregate.shuffleLocalBytesRead`	Die Anzahl der Byte, die seit dem letzten Bericht von allen Executoren gelesen wurden, um Daten zwischen ihnen zu mischen (vom AWS Glue Metrics Dashboard als die Anzahl der zu diesem Zweck in der letzten Minute gelesenen Byte aggregiert). Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Einheit: Byte Kann verwendet werden, um Folgendes zu überwachen: Datenmischung in Aufträgen (große Joins, GroupBy, Repartition, Coalesce). Einige Möglichkeiten, die Daten zu verwenden: • Repartitionieren oder Dekomprimieren Sie große Eingabedateien vor der weiteren Verarbeitung. • Repartitionieren Sie Daten einheitlicher mit Hotkeys. • Vorfiltern von Daten vor Joins oder GroupBy-Operationen.

Metrik	Beschreibung
`glue.driver.BlockManager.disk.diskSpaceUsed_MB`	Die Anzahl der Megabyte an Speicherplatz, die für alle Executoren verwendet werden. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge). Gültige Statistiken: Durchschnitt. Dies ist eine Spark-Metrik, die als absoluter Wert gemeldet wird. Einheit: Megabyte Kann für die Überwachung folgender Aspekte verwendet werden: • Festplattenspeicher, der für Blöcke verwendet wird, die zwischengespeicherte RDD-Partitionen darstellen. • Festplattenspeicher, der für Blöcke verwendet wird, die Zwischenausgaben für Shuffle darstellen. • Festplattenspeicher, der für Blöcke verwendet wird, die Broadcasts darstellen. Einige Möglichkeiten, die Daten zu verwenden: • Identifizieren von Auftragsfehlern aufgrund erhöhter Festplattenauslastung. • Identifizieren Sie große Partitionen, die zum Fluten oder Mischen führen. • Erhöhen Sie die bereitgestellte DPU-Kapazität, um diese Probleme zu beheben.

Metrik	Beschreibung
`glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`	Die Anzahl der aktiven Auftrags-Executors. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge). Gültige Statistiken: Durchschnitt. Dies ist eine Spark-Metrik, die als absoluter Wert gemeldet wird. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Auftragsaktivität. • Straggling-Executors (die nur mit ein paar Executors laufen) • Aktuelle Parallelität auf Executor-Ebene. Einige Möglichkeiten, die Daten zu verwenden: • Repartitionieren oder Dekomprimieren großer Eingabedateien im Voraus, wenn Cluster nicht ausgelastet ist. • Identifizieren Sie Verzögerungen bei der Ausführung von Phasen oder Aufträgen aufgrund von Straggler-Szenarien. • • Vergleichen Sie mit numberMaxNeeded Executors, um mehr über den Backlog bei der Bereitstellung zu erfahren. DPUs

Metrik	Beschreibung
`glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`	Die Anzahl der maximalen (aktiv ausgeführten und ausstehenden) Auftrags-Executors, die benötigt werden, um die aktuelle Last zu erfüllen. Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge). Gültige Statistiken: Maximum. Dies ist eine Spark-Metrik, die als absoluter Wert gemeldet wird. Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Auftragsaktivität. • Aktuelle Parallelität auf Executor-Ebene und Rückstand ausstehender Aufgaben, die wegen nicht verfügbaren Executors aufgrund von DPU-Kapazität oder beendeten/fehlgeschlagenen Executors noch nicht geplant wurden. Einige Möglichkeiten, die Daten zu verwenden: • Identifizieren Sie den Ausstand/Rückstand der Scheduling-Warteschlange. • Identifizieren Sie Verzögerungen bei der Ausführung von Phasen oder Aufträgen aufgrund von Straggler-Szenarien. • Vergleichen Sie mit numberAllExecutors , um mehr über den Backlog bei der Bereitstellung zu erfahren. DPUs

Metrik	Beschreibung
	• Erhöhen Sie die bereitgestellte DPU-Kapazität, um den ausstehenden Executor-Backlog zu korrigier en.

Metrik	Beschreibung

`glue.driver.jvm.heap.usage`
 	
`glue.executorId.jvm.heap.usage`
 	
`glue.ALL.jvm.heap.usage`
	Den vom JVM-Heap für diesen Treiber (Skalierung: 0-1) verwendeten Speicheranteil – vom Treiber, einem durch executorId identifizierten Executor oder ALLE Executors.
	
	Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge).
	
	Gültige Statistiken: Durchschnitt. Dies ist eine Spark-Metrik, die als absoluter Wert gemeldet wird.
	
	Einheit: Prozentsatz
	
	Kann für die Überwachung folgender Aspekte verwendet werden:
	
	• out-of-memoryTreiberbedingungen (OOM) verwenden`glue.driver.jvm.heap.usage` .
	• out-of-memoryExecutor-Bedingungen (OOM) verwenden. `glue.ALL.jvm.heap.usage`
	
	Einige Möglichkeiten, die Daten zu verwenden:
	
	• Identifizieren Sie speicheraufwändige Executor-IDs und -Phasen.
	• Identifizieren Sie Straggling-Executor-IDs und -Phasen.
	• Identifizieren Sie einen out-of-memory Treiberzu stand (OOM).
	• Identifizieren Sie eine out-of-memory Executor-Bedingung (OOM) und rufen Sie die entsprechende Executor-ID ab, um einen Stack-Trace aus dem Executor-Log abrufen zu können.

Metrik	Beschreibung
	• Identifizieren Sie Dateien oder Partitionen, die Datenverzerrungen aufweisen können, die zu Nachzüglern oder Bedingungen führen können (). out-of-memory OOMs

Metrik	Beschreibung

`glue.driver.jvm.heap.used`
 	
`glue.executorId.jvm.heap.used`
 	
`glue.ALL.jvm.heap.used`
	Die Anzahl der vom JVM-Heap verwendeten Speicherbytes für den Treiber, der durch executorId oder ALLE Executors identifiziert wurde.
	
	Gültige Dimensionen: `JobName` (der Name des AWS Glue Job), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge).
	
	Gültige Statistiken: Durchschnitt. Dies ist eine Spark-Metrik, die als absoluter Wert gemeldet wird.
	
	Einheit: Byte
	
	Kann für die Überwachung folgender Aspekte verwendet werden:
	
	• out-of-memoryFahrerbedingungen (OOM).
	• out-of-memoryBedingungen für den Executor (OOM).
	
	Einige Möglichkeiten, die Daten zu verwenden:
	
	• Identifizieren Sie speicheraufwändige Executor-IDs und -Phasen.
	• Identifizieren Sie Straggling-Executor-IDs und -Phasen.
	• Identifizieren Sie einen out-of-memory Treiberzu stand (OOM).
	• Identifizieren Sie eine out-of-memory Executor-Bedingung (OOM) und rufen Sie die entsprech ende Executor-ID ab, um einen Stack-Trace aus dem Executor-Log abrufen zu können.
	• Identifizieren Sie Dateien oder Partitionen, die Datenverzerrungen aufweisen können, die zu

Metrik	Beschreibung
	Nachzüglern oder Bedingungen führen können (). out-of-memory OOMs

Metrik	Beschreibung

`glue.driver.s3.filesystem.read_bytes`
 	
`glue.executorId.s3.filesystem.read_bytes`
 	
`glue.ALL.s3.filesystem.read_bytes`
	Die Anzahl der Byte, die der Treiber, ein durch die ExecutorID identifizierter Executor oder ALLE Executoren seit dem letzten Bericht aus Amazon S3 gelesen haben (aggregiert vom AWS Glue Metrics Dashboard als Anzahl der in der letzten Minute gelesenen Byte).
	
	Gültige Dimensionen: `JobName`, `JobRunId` und `Type` (Messung).
	
	Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Der Bereich unter der Kurve im AWS Glue Metrics Dashboard kann verwendet werden, um die von zwei verschiedenen Jobläufen gelesenen Bytes visuell zu vergleichen.
	
	Einheit: Byte.
	
	Kann für die Überwachung folgender Aspekte verwendet werden:
	
	• ETL-Datenbewegung.
	• Fortschritt des Auftrags.
	• Probleme mit Auftragslesezeichen (verarbeitete, wiederverarbeitete und übersprungene Daten).
	• Vergleich der Lesevorgänge mit der Erfassung srate aus externen Datenquellen.
	• Abweichung über Auftragsausführungen hinweg.
	
	Die resultierenden Daten können verwendet werden für:

Metrik	Beschreibung
	• DPU-Kapazitätsplanung. • Einstellen von Alarmen für große Daten-Spikes oder Einbrüche, die für Auftragsausführungen und Auftragsphasen gelesen werden.

Metrik	Beschreibung

`glue.driver.s3.filesystem.write_bytes`
 	
`glue.executorId.s3.filesystem.write_bytes`
 	
`glue.ALL.s3.filesystem.write_bytes`
	Die Anzahl der Byte, die der Treiber, ein durch die ExecutorID identifizierter Executor oder ALLE Executoren seit dem letzten Bericht in Amazon S3 geschrieben haben (aggregiert vom AWS Glue Metrics Dashboard als Anzahl der in der letzten Minute geschriebenen Byte).
	
	Gültige Dimensionen: `JobName`, `JobRunId` und `Type` (Messung).
	
	Gültige Statistiken: Summe Bei dieser Metrik handelt es sich um einen Deltawert gegenüber dem zuletzt gemeldeten Wert. Daher wird im AWS Glue Metrik-Dashboard eine SUM-Statistik für die Aggregation verwendet. Der Bereich unter der Kurve im AWS Glue Metrik-Dashboard kann verwendet werden, um die in zwei verschiedenen Jobläufen geschriebenen Bytes visuell zu vergleichen.
	
	Einheit: Byte
	
	Kann für die Überwachung folgender Aspekte verwendet werden:
	
	• ETL-Datenbewegung.
	• Fortschritt des Auftrags.
	• Probleme mit Auftragslesezeichen (verarbeitete, wiederverarbeitete und übersprungene Daten).
	• Vergleich der Lesevorgänge mit der Erfassung srate aus externen Datenquellen.
	• Abweichung über Auftragsausführungen hinweg.
	
	Einige Möglichkeiten, die Daten zu verwenden:
	
	• DPU-Kapazitätsplanung.

Metrik	Beschreibung
	• Einstellen von Alarmen für große Daten-Spikes oder Einbrüche, die für Auftragsausführungen und Auftragsphasen gelesen werden.
`glue.driver.streaming.numRecords`	Die Anzahl der Datensätze, die in einem Mikrobatch empfangen werden. Diese Metrik ist nur für AWS Glue Streaming-Jobs mit AWS Glue Version 2.0 und höher verfügbar. Gültige Dimensionen: `JobName` (der Name des AWS Glue Jobs), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe, Maximum, Minimum, Durchschnitt, Prozent Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Gelesene Datensätze. • Fortschritt des Auftrags.

Metrik	Beschreibung
`glue.driver.streaming.batchProcessingTimeInMs`	Die Zeit, die für die Verarbeitung der Batches in Millisekunden benötigt wird. Diese Metrik ist nur für AWS Glue Streaming-Jobs mit AWS Glue Version 2.0 und höher verfügbar. Gültige Dimensionen: `JobName` (der Name des AWS Glue Jobs), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Anzahl). Gültige Statistiken: Summe, Maximum, Minimum, Durchschnitt, Prozent Einheit: Anzahl Kann für die Überwachung folgender Aspekte verwendet werden: • Fortschritt des Auftrags. • Skript-Leistung.

Metrik	Beschreibung

`glue.driver.system.cpuSystemLoad`
 	
`glue.executorId.system.cpuSystemLoad`
 	
`glue.ALL.system.cpuSystemLoad`
	Der vom Treiber verwendete Anteil der CPU-Systemauslastung (Skalierung: 0-1) – ein durch executorId identifizierter Executor oder ALLE Executors.
	
	Gültige Dimensionen: `JobName` (der Name des AWS Glue Jobs), `JobRunId` (die JobRun ID. oder`ALL`) und `Type` (Gauge).
	
	Gültige Statistiken: Durchschnitt. Diese Metrik wird als absoluter Wert gemeldet.
	
	Einheit: Prozentsatz
	
	Kann für die Überwachung folgender Aspekte verwendet werden:
	
	• CPU-Auslastung des Treibers.
	• CPU-Auslastung des Executors.
	• Erkennen von CPU-gebundenen oder E/A-gebundenen Executors oder Phasen in einem Auftrag.
	
	Einige Möglichkeiten, die Daten zu verwenden:
	
	• DPU-Kapazitätsplanung zusammen mit E/A-Metriken (gelesene Bytes/Shuffle-Bytes, Task-Parallelismus) und der Anzahl der maximal benötigten Executor-Metrik.
	• Identifizieren Sie das CPU/E/A-gebundene Verhältnis. Dies ermöglicht eine Neupartitionierung und Erhöhung der bereitgestellten Kapazität für Aufträge mit langer Laufzeit mit aufteilbaren Datensätzen mit einer geringeren CPU-Auslastung.

Dimensionen für Metriken AWS Glue

AWS Glue Metriken verwenden den AWS Glue Namespace und stellen Metriken für die folgenden Dimensionen bereit:

Dimension	Beschreibung
`JobName`	Diese Dimension filtert nach Metriken aller Jobausführungen eines bestimmten AWS Glue Jobs.
`JobRunId`	Diese Dimension filtert nach Metriken eines bestimmten AWS Glue Jobs, der von einer JobRun ID ausgeführt wird, oder`ALL`.
`Type`	Diese Dimension filtert nach Metriken entweder nach `count` (eine aggregierte Zahl) oder `gauge` (ein Wert zu einem bestimmten Zeitpunkt).

Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

CloudWatch Amazon-Alarme einrichten AWS Glue Berufsprofile

AWS Glue Metriken sind auch bei Amazon verfügbar CloudWatch. Sie können Alarme für jeden einrichten AWS Glue Metrik für geplante Jobs.

Einige häufige Szenarien für das Einrichten von Alarmen:

- Jobs, denen der Arbeitsspeicher ausgeht (OOM): Stellen Sie einen Alarm ein, wenn die Speichernutzung den normalen Durchschnitt für den Treiber oder einen Executor für einen AWS Glue Job.
- Nachzügler: Stellen Sie einen Alarm ein, wenn die Anzahl der Testamentsvollstrecker innerhalb eines Zeitraums über einen längeren Zeitraum unter einen bestimmten Schwellenwert fällt AWS Glue Job.
- Datenstau oder Wiederverarbeitung: Vergleichen Sie die Metriken einzelner Jobs in einem Workflow mithilfe eines CloudWatch mathematischen Ausdrucks. Sie können dann einen Alarm für den resultierenden Ausdruckswert auslösen (z. B. das Verhältnis der von einem Auftrag geschriebenen Bytes zu den von einem nachfolgenden Auftrag gelesenen Bytes).

Detaillierte Anweisungen zum Einstellen von Alarmen finden Sie unter Einen CloudWatch Alarm erstellen oder bearbeiten im Amazon CloudWatch Events-Benutzerhandbuch.

Informationen zum Überwachen und Debuggen von Szenarien mit Hilfe von CloudWatch finden Sie unterAuftragsüberwachung und Debugging.

Protokollierung für AWS Glue Jobs

In AWS Glue 5.0 verfügen alle Jobs über Funktionen zur Echtzeitprotokollierung. Darüber hinaus können Sie benutzerdefinierte Konfigurationsoptionen angeben, um das Protokollierungsverhalten anzupassen. Zu diesen Optionen gehören das Festlegen des Amazon CloudWatch Protokollgruppennamens, des Amazon CloudWatch Protokollstream-Präfixes (das der ID und driver/executor ID der AWS Glue Auftragsausführung vorangestellt wird) und des Protokollkonvertierungsmusters für Protokollnachrichten. Mit diesen Konfigurationen können Sie Protokolle in benutzerdefinierten Amazon CloudWatch Protokollgruppen mit unterschiedlichen Ablaufrichtlinien zusammenfassen. Darüber hinaus können Sie die Protokolle effektiver analysieren, indem Sie benutzerdefinierte Protokollstream-Präfixe und Konvertierungsmuster verwenden. Dieser Grad der Anpassung ermöglicht es Ihnen, die Protokollverwaltung und -analyse gemäß Ihren spezifischen Anforderungen zu optimieren.

Protokollierungsverhalten in AWS Glue 5.0

Standardmäßig werden Systemprotokolle, Spark-Daemon-Protokolle und AWS Glue Benutzer-Logger-Protokolle in Amazon CloudWatch die `/aws-glue/jobs/error` Protokollgruppe geschrieben. Andererseits werden die Benutzerprotokolle stdout (Standardausgabe) und stderr (Standardfehler) standardmäßig in die `/aws-glue/jobs/output` Protokollgruppe geschrieben.

Benutzerdefiniertes Logging

Sie können die Standardpräfixe für Protokollgruppen und Protokolldatenströme mithilfe der folgenden Job-Argumente anpassen:

- `--custom-logGroup-prefix`: Ermöglicht es Ihnen, ein benutzerdefiniertes Präfix für die Gruppen `/aws-glue/jobs/error` und `/aws-glue/jobs/output` Logs anzugeben. Wenn Sie ein benutzerdefiniertes Präfix angeben, haben die Namen der Protokollgruppen das folgende Format:
  - `/aws-glue/jobs/error` wird *`<customer prefix>`*`/error` sein.
  - `/aws-glue/jobs/output` wird *`<customer prefix>`*`/output` sein.

- `--custom-logStream-prefix`: Ermöglicht es Ihnen, ein benutzerdefiniertes Präfix für die Log-Stream-Namen innerhalb der Log-Gruppen anzugeben. Wenn Sie ein benutzerdefiniertes Präfix angeben, haben die Log-Stream-Namen das folgende Format:
  - `jobrunid-driver` wird *`<customer log stream>`*`-driver` sein.
  - `jobrunid-executorNum` wird *`<customer log stream>`*`-executorNum` sein.

Validierungsregeln und Einschränkungen für benutzerdefinierte Präfixe:

- Der gesamte Protokollstreamname muss zwischen 1 und 512 Zeichen lang sein.
- Das benutzerdefinierte Präfix selbst ist auf 400 Zeichen beschränkt.
- Das benutzerdefinierte Präfix muss dem regulären Ausdrucksmuster `[^: *] *` entsprechen (zulässige Sonderzeichen sind '_', '-' und '/').

Protokollieren von anwendungsspezifischen Meldungen mit dem benutzerdefinierten Skript-Logger

Sie können den AWS Glue Logger verwenden, um alle anwendungsspezifischen Meldungen im Skript zu protokollieren, die in Echtzeit an den Treiberprotokollstream gesendet werden.

Das folgende Beispiel zeigt ein Python-Skript.

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
logger = glueContext.get_logger()
logger.info("info message")
logger.warn("warn message")
logger.error("error message")
```

Das folgende Beispiel zeigt ein Scala-Skript.

```
import com.amazonaws.services.glue.log.GlueLogger

object GlueApp {
  def main(sysArgs: Array[String]) {
    val logger = new GlueLogger
    logger.info("info message")
    logger.warn("warn message")
```

```
    logger.error("error message")
  }
}
```

Aktivieren des Fortschrittsbalkens zum Anzeigen des Auftragsfortschritts

AWS Glue stellt unter dem `JOB_RUN_ID-progress-bar` Protokollstream eine Fortschrittsleiste in Echtzeit bereit, mit der der Status der AWS Glue Auftragsausführung überprüft werden kann. Derzeit unterstützt er nur Aufträge, die `glueContext` initialisieren. Wenn Sie einen reinen Spark-Job ohne Initialisierung ausführen`glueContext`, wird der AWS Glue Fortschrittsbalken nicht angezeigt.

Der Fortschrittsbalken wird alle 5 Sekunden aktualisiert.

```
Stage Number (Stage Name): > (numCompletedTasks + numActiveTasks) /
 totalNumOfTasksInThisStage]
```

Sicherheitskonfiguration mit Protokollierung Amazon CloudWatch

Wenn eine Sicherheitskonfiguration für Amazon CloudWatch Protokolle aktiviert ist, werden Protokollgruppen mit bestimmten Benennungsmustern AWS Glue erstellt, die den Namen der Sicherheitskonfiguration enthalten.

Benennung von Protokollgruppen mit Sicherheitskonfiguration

Die Standard- und benutzerdefinierten Protokollgruppen lauten wie folgt:

- Standard-Fehlerprotokollgruppe: `/aws-glue/jobs/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/error`
- Standardgruppe für das Ausgabelogbuch: `/aws-glue/jobs/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/output`
- Benutzerdefinierte Fehlerprotokollgruppe (AWS Glue 5.0): *`custom-log-group-prefix`*`/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/error`
- Benutzerdefinierte Ausgabeloggruppe (AWS Glue 5.0): *`custom-log-group-prefix`*`/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/output`

Erforderliche IAM-Berechtigungen

Sie müssen die `logs:AssociateKmsKey` Berechtigung zu Ihren IAM-Rollenberechtigungen hinzufügen, wenn Sie eine Sicherheitskonfiguration mit Amazon CloudWatch Logs aktivieren. Wenn diese Berechtigung nicht enthalten ist, wird die fortlaufende Protokollierung deaktiviert.

Um die Verschlüsselung für die Amazon CloudWatch Logs zu konfigurieren, folgen Sie außerdem den Anweisungen unter Log Data in Amazon CloudWatch Logs Using verschlüsseln AWS Key Management Service im Amazon Amazon CloudWatch Logs-Benutzerhandbuch.

Zusätzliche Informationen

Weitere Informationen zum Erstellen von Sicherheitskonfigurationen finden Sie unter Sicherheitskonfigurationen auf der AWS Glue Konsole verwalten.

Themen

- Kontinuierliche Protokollierung für Jobs der AWS Glue Version 4.0 und früher aktivieren
- Protokolle für AWS Glue Jobs anzeigen

Kontinuierliche Protokollierung für Jobs der AWS Glue Version 4.0 und früher aktivieren

> **Note**
>
> In AWS Glue 4.0 und früheren Versionen war die kontinuierliche Protokollierung eine verfügbare Funktion. Mit der Einführung von AWS Glue 5.0 können jedoch alle Jobs in Echtzeit protokolliert werden. Weitere Informationen zu den Protokollierungsfunktionen und Konfigurationsoptionen in AWS Glue 5.0 finden Sie unter Protokollierung für AWS Glue Jobs.

Sie können die kontinuierliche Protokollierung über die AWS Glue Konsole oder über AWS Command Line Interface (AWS CLI) aktivieren.

Sie können die kontinuierliche Protokollierung aktivieren, wenn Sie einen neuen Job erstellen, einen vorhandenen Job bearbeiten oder ihn über den aktivieren AWS CLI.

Sie können auch benutzerdefinierte Konfigurationsoptionen angeben, z. B. den Namen der Amazon CloudWatch Protokollgruppe, das CloudWatch Protokollstream-Präfix vor der driver/executor ID der AWS Glue Auftragsausführung und das Protokollkonvertierungsmuster für Protokollnachrichten. Diese Konfigurationen helfen Ihnen dabei, aggregierte Protokolle in benutzerdefinierten CloudWatch Protokollgruppen mit unterschiedlichen Ablaufrichtlinien einzurichten und sie mit benutzerdefinierten Logstream-Präfixen und Konvertierungsmustern weiter zu analysieren.

Themen

- Unter Verwendung der AWS Management Console

- Protokollieren von anwendungsspezifischen Meldungen mit dem benutzerdefinierten Skript-Logger
- Aktivieren des Fortschrittsbalkens zum Anzeigen des Auftragsfortschritts
- Sicherheitskonfiguration mit kontinuierlicher Protokollierung

Unter Verwendung der AWS Management Console

Gehen Sie wie folgt vor, um mithilfe der Konsole die kontinuierliche Protokollierung beim Erstellen oder Bearbeiten eines AWS Glue Jobs zu aktivieren.

Um einen neuen AWS Glue Job mit kontinuierlicher Protokollierung zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option ETL-Jobs aus.
3. Wählen Sie Visual ETL.
4. Erweitern Sie auf der Registerkarte Jobdetails den Abschnitt Erweiterte Eigenschaften.
5. Wählen Sie unter Kontinuierliche Protokollierung die Option Anmeldungen aktivieren aus CloudWatch.

Um die kontinuierliche Protokollierung für einen vorhandenen AWS Glue Job zu aktivieren

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
3. Wählen Sie einen Auftrag in der Liste Jobs (Aufträge) aus.
4. Wählen Sie Action (Aktion), Edit job (Auftrag bearbeiten) aus.
5. Erweitern Sie auf der Registerkarte Jobdetails den Abschnitt Erweiterte Eigenschaften.
6. Wählen Sie unter Kontinuierliche Protokollierung die Option Anmeldungen aktivieren aus CloudWatch.

Verwenden des AWS CLI

Um die kontinuierliche Protokollierung zu aktivieren, übergeben Sie Jobparameter an einen AWS Glue Job. Übergeben Sie die folgenden speziellen Job-Parameter, die anderen AWS Glue Job-Parametern ähneln. Weitere Informationen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

```
'--enable-continuous-cloudwatch-log': 'true'
```

Sie können einen benutzerdefinierten CloudWatch Amazon-Protokollgruppennamen angeben. Sofern nicht angegeben, lautet der Standardname der Protokollgruppe `/aws-glue/jobs/logs-v2`.

```
'--continuous-log-logGroup': 'custom_log_group_name'
```

Sie können ein benutzerdefiniertes CloudWatch Amazon-Logstream-Präfix angeben. Sofern nicht angegeben, ist das standardmäßige Logstream-Präfix die Auftragsausführungs-ID.

```
'--continuous-log-logStreamPrefix': 'custom_log_stream_prefix'
```

Sie können ein benutzerdefiniertes Konvertierungsmuster für die kontinuierliche Protokollierung angeben. Sofern nicht angegeben, ist das Standardkonvertierungsmuster `%d{yy/MM/dd HH:mm:ss} %p %c{1}: %m%n`. Beachten Sie, dass das Konvertierungsmuster nur für Treiberprotokolle und Executor-Protokolle gilt. Dies wirkt sich nicht auf die AWS Glue - Fortschrittsleiste aus.

```
'--continuous-log-conversionPattern': 'custom_log_conversion_pattern'
```

Protokollieren von anwendungsspezifischen Meldungen mit dem benutzerdefinierten Skript-Logger

Sie können den AWS Glue Logger verwenden, um alle anwendungsspezifischen Meldungen im Skript zu protokollieren, die in Echtzeit an den Treiberprotokollstream gesendet werden.

Das folgende Beispiel zeigt ein Python-Skript.

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
logger = glueContext.get_logger()
logger.info("info message")
logger.warn("warn message")
logger.error("error message")
```

Das folgende Beispiel zeigt ein Scala-Skript.

```
import com.amazonaws.services.glue.log.GlueLogger
```

```
object GlueApp {
  def main(sysArgs: Array[String]) {
    val logger = new GlueLogger
    logger.info("info message")
    logger.warn("warn message")
    logger.error("error message")
  }
}
```

Aktivieren des Fortschrittsbalkens zum Anzeigen des Auftragsfortschritts

AWS Glue stellt unter dem `JOB_RUN_ID-progress-bar` Protokollstream eine Fortschrittsleiste in Echtzeit bereit, mit der der Status der AWS Glue Auftragsausführung überprüft werden kann. Derzeit unterstützt er nur Aufträge, die `glueContext` initialisieren. Wenn Sie einen reinen Spark-Job ohne Initialisierung ausführen`glueContext`, wird der AWS Glue Fortschrittsbalken nicht angezeigt.

Der Fortschrittsbalken wird alle 5 Sekunden aktualisiert.

```
Stage Number (Stage Name): > (numCompletedTasks + numActiveTasks) /
 totalNumOfTasksInThisStage]
```

Sicherheitskonfiguration mit kontinuierlicher Protokollierung

Wenn eine Sicherheitskonfiguration für CloudWatch Protokolle aktiviert ist, AWS Glue wird eine Protokollgruppe mit dem folgenden Namen für fortlaufende Protokolle erstellt:

```
<Log-Group-Name>-<Security-Configuration-Name>
```

Die Standard- und benutzerdefinierten Protokollgruppen lauten wie folgt:

- Die standardmäßige fortlaufende Protokollgruppe lautet `/aws-glue/jobs/error-`*<Security-Configuration-Name>*
- Die benutzerdefinierte fortlaufende Protokollgruppe lautet *<custom-log-group-name>*-*<Security-Configuration-Name>*

Sie `logs:AssociateKmsKey` müssen Ihrer IAM-Rollenberechtigungen hinzufügen, wenn Sie eine Sicherheitskonfiguration mit CloudWatch Protokollen aktivieren. Wenn diese Berechtigung nicht enthalten ist, wird die fortlaufende Protokollierung deaktiviert. Um die Verschlüsselung für

die CloudWatch Logs zu konfigurieren, folgen Sie außerdem den Anweisungen unter Log Data in CloudWatch Logs Using verschlüsseln AWS Key Management Service im Amazon CloudWatch Logs-Benutzerhandbuch.

Weitere Informationen zum Erstellen einer Sicherheitskonfiguration finden Sie hier: Sicherheitskonfigurationen auf der AWS Glue Konsole verwalten.

> **Note**
>
> Es können zusätzliche Gebühren anfallen, wenn Sie die Protokollierung aktivieren und zusätzliche CloudWatch Protokollereignisse erstellt werden. Weitere Informationen finden Sie unter CloudWatch Amazon-Preise.

Protokolle für AWS Glue Jobs anzeigen

Sie können Echtzeitprotokolle über die AWS Glue Konsole oder die CloudWatch Amazon-Konsole anzeigen.

Um Echtzeitprotokolle über das AWS Glue Konsolen-Dashboard anzuzeigen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option Jobs (Aufträge) aus.
3. Fügen Sie einen Auftrag hinzu oder starten Sie einen vorhandenen Auftrag. Wählen Sie Action (Aktion), Run job (Auftrag ausführen) aus.

   Wenn Sie einen Auftrag starten, navigieren Sie zu einer Seite mit Informationen über den laufenden Auftrag:

   - Die Registerkarte Logs (Protokolle) zeigt die älteren aggregierten Anwendungsprotokolle an.
   - Auf der Registerkarte Protokolle wird eine Fortschrittsleiste in Echtzeit angezeigt, wenn der Job mit `glueContext` initialized ausgeführt wird.
   - Die Registerkarte Protokolle enthält auch die Treiberprotokolle, die Apache Spark-Treiberprotokolle in Echtzeit erfassen, und Anwendungsprotokolle aus dem Skript, das mit dem AWS Glue Anwendungslogger protokolliert wurde, wenn der Job ausgeführt wird.
4. Für ältere Aufträge können Sie auch die Echtzeitprotokolle unter Job History (Auftragsverlauf) anzeigen, indem Sie Jobs (Aufträge) auswählen. Mit dieser Aktion gelangen Sie zu der

CloudWatch Konsole, auf der alle Protokollstreams für Spark-Treiber, Executor und Fortschrittsbalken für diese Jobausführung angezeigt werden.

Um Echtzeitprotokolle über das CloudWatch Konsolen-Dashboard anzuzeigen

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Log (Protokoll) aus.
3. Wählen Sie die Protokollgruppe /aws-glue/jobs/error/aus.
4. Fügen Sie die Auftragsausführung-ID in das Feld Filter ein.

   Sie können die Treiberprotokolle, die Executor-Protokolle und den Fortschrittsbalken anzeigen (wenn der Standard filter (Standardfilter) verwendet wird).

Überwachen mit AWS Glue Beobachtbarkeitsmetriken

> **Note**
>
> AWS Glue Messwerte zur Beobachtbarkeit sind verfügbar unter AWS Glue 4.0 und spätere Versionen.

Verwenden Sie AWS Glue Messwerte zur Beobachtbarkeit, um Einblicke in das zu gewinnen, was in Ihrem Inneren passiert AWS Glue für Apache Spark-Jobs, um die Triaging und Analyse von Problemen zu verbessern. Beobachtbarkeitsmetriken werden über Amazon CloudWatch - Dashboards visualisiert und können verwendet werden, um die Ursache von Fehlern zu analysieren und Leistungsengpässe zu diagnostizieren. Sie können den Zeitaufwand für das Debuggen von Problemen reduzieren, sodass Sie sich darauf konzentrieren können, Probleme schneller und effektiver zu lösen.

AWS Glue Observability bietet Amazon CloudWatch Metriken, die in die folgenden vier Gruppen unterteilt sind:

- Zuverlässigkeit (d. h. Fehlerklassen) – Identifizieren Sie ganz einfach die häufigsten Fehlerursachen in einem bestimmten Zeitraum, die Sie möglicherweise beheben möchten.
- Leistung (d. h. Schiefe) – Identifizieren Sie Leistungsengpässe und wenden Sie Optimierungsmethoden an. Wenn Sie beispielsweise Leistungseinbußen aufgrund von

Auftragsschiefe feststellen, können Sie die adaptive Abfrageausführung von Spark aktivieren und den Schwellenwert für schiefe Verknüpfungen anpassen.

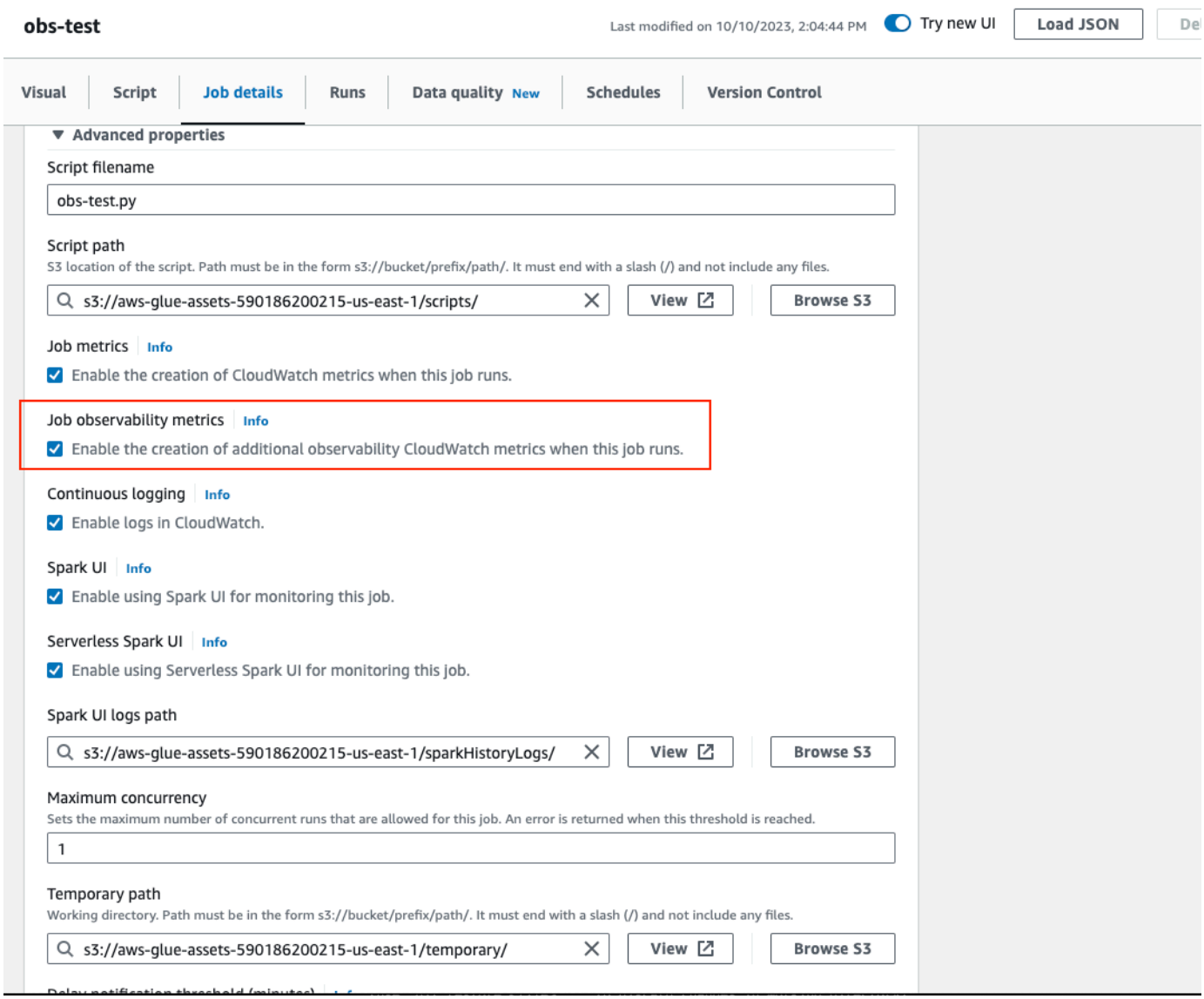
- Durchsatz (d. h. Durchsatz pro Quelle/Senke) – Überwachen Sie Trends bei Lese- und Schreibvorgängen. Sie können auch Amazon CloudWatch Alarme für Anomalien konfigurieren.
- Ressourcenauslastung (d. h. Mitarbeiter, Speicher- und Festplattennutzung) – Lokalisieren Sie Aufträge mit geringer Kapazitätsauslastung auf effiziente Weise. Möglicherweise möchten Sie Folgendes aktivieren AWS Glue auto-scaling für diese Jobs.

Erste Schritte mit AWS Glue Beobachtbarkeitsmetriken

> **Note**
>
> Die neuen Metriken sind standardmäßig aktiviert in AWS Glue Studio console.

Um Beobachtbarkeitsmetriken zu konfigurieren in AWS Glue Studio:

1. Loggen Sie sich ein bei AWS Glue Konsole und wählen Sie im Konsolenmenü ETL-Jobs aus.
2. Klicken Sie im Bereich Ihre Aufträge auf den Namen des gewünschten Auftrags.
3. Wählen Sie die Registerkarte Job details (Auftragsdetails) aus.
4. Scrollen Sie nach unten, wählen Sie Erweiterte Eigenschaften und dann Metriken zur Auftragsbeobachtbarkeit aus.



Um zu aktivieren AWS Glue Messwerte zur Beobachtbarkeit unter Verwendung von AWS CLI:

- Fügen Sie der `--default-arguments`-Zuordnung in der JSON-Eingabedatei den folgenden Schlüsselwert hinzu:

```
--enable-observability-metrics, true
```

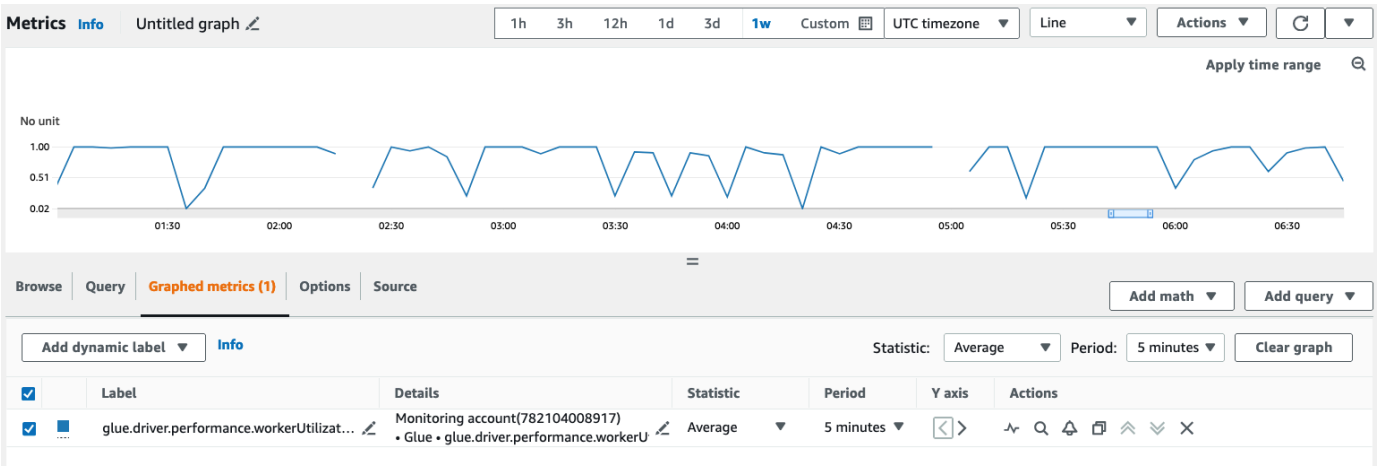
Die Verwendung von AWS Glue observability

Weil die AWS Glue Observability-Metriken werden über bereitgestellt. Sie können die Amazon CloudWatch Konsole Amazon CloudWatch, das SDK oder die API verwenden AWS CLI, um die Datenpunkte der Observability-Metriken abzufragen. Ein Beispiel für einen Anwendungsfall finden Sie unter Verwenden von Glue Observability zur Überwachung der Ressourcennutzung zur Kostensenkung, wann AWS Glue Messwerte zur Beobachtbarkeit.

Die Verwendung von AWS Glue Beobachtbarkeit in der Konsole Amazon CloudWatch

Um Metriken in der Amazon CloudWatch Konsole abzufragen und zu visualisieren:

1. Öffnen Sie die Amazon CloudWatch Konsole und wählen Sie Alle Metriken aus.
2. Wählen Sie unter Benutzerdefinierte Namespaces AWS Glue.
3. Wählen Sie „Metriken zur Auftragsbeobachtbarkeit“, „Beobachtbarkeitsmetriken pro Quelle“ oder „Beobachtbarkeitsmetriken pro Senke“ aus.
4. Suchen Sie nach dem gewünschten Metrik- und Auftragsnamen sowie der Auftragsausführungs-ID und wählen Sie sie aus.
5. Konfigurieren Sie auf der Registerkarte Grafische Metriken Ihre bevorzugte Statistik, den Zeitraum und andere Optionen.



Um eine Observability-Metrik abzufragen, verwenden Sie AWS CLI:

1. Erstellen Sie eine JSON-Datei mit einer Metrikdefinition und ersetzen Sie `your-Glue-job-name` und `your-Glue-job-run-id` mit Ihren Werten.

```
$ cat multiplequeries.json
```

```
[
    {
        "Id": "avgWorkerUtil_0",
        "MetricStat": {
            "Metric": {
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-A>"
                    },
                    {
                        "Name": "JobRunId",
                        "Value": "<your-Glue-job-run-id-A>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
                    },
                    {
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    },
    {
        "Id": "avgWorkerUtil_1",
        "MetricStat": {
            "Metric": {
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-B>"
                    },
                    {
                        "Name": "JobRunId",
```

```
                        "Value": "<your-Glue-job-run-id-B>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
                    },
                    {
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    }
]
```

2. Führen Sie den Befehl `get-metric-data` aus:

```
$ aws cloudwatch get-metric-data --metric-data-queries file: //multiplequeries.json
 \
     --start-time '2023-10-28T18: 20' \
     --end-time '2023-10-28T19: 10'  \
     --region us-east-1
{
    "MetricDataResults": [
        {
            "Id": "avgWorkerUtil_0",
            "Label": "<your-label-for-A>",
            "Timestamps": [
                "2023-10-28T18:20:00+00:00"
            ],
            "Values": [
                0.06718750000000001
            ],
            "StatusCode": "Complete"
        },
        {
            "Id": "avgWorkerUtil_1",
            "Label": "<your-label-for-B>",
```

```
            "Timestamps": [
                "2023-10-28T18:50:00+00:00"
            ],
            "Values": [
                0.5959183673469387
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

Beobachtbarkeitsmetriken

AWS Glue Observability erstellt Amazon CloudWatch alle 30 Sekunden ein Profil der folgenden Metriken und sendet sie. Einige dieser Metriken können in der AWS Glue Studio Seite „Überwachung von Auftragsausführungen“.

Metrik	Beschreibung	Kategorie
glue.driver.skewness.stage	Metrikkategorie: job_performance Abweichungen bei der Ausführung von Spark-Phasen: Diese Metrik erfasst Abweichungen bei der Ausführung, die durch schiefe Eingabedaten oder durch eine Transformation (z. B. schiefe Verknüpfung) verursacht werden. Die Werte dieser Metrik fallen in den Bereich [0, unendlich[, wobei 0 bedeutet, dass der Unterschied zwischen der maximalen und der mittleren Ausführungszeit von Aufgaben in der	job_performance

Metrik	Beschreibung	Kategorie
	Phase weniger beträgt als ein bestimmter Phasenabweichungsfaktor. Der Standardabweichungsfaktor von Phasen ist „5“ und wird mit dieser Spark-Konfiguration überschrieben: spark.metrics.conf.driver.source.glue.jobPerformance.skewnessFactor Ein Wert von 1 für die Phasenabweichung bedeutet, dass der Unterschied das Doppelte des Abweichungsfaktors der Phase beträgt. Der Wert der Stufenschiefe wird alle 30 Sekunden aktualisiert, um die aktuelle Schiefe widerzuspiegeln. Der Wert am Ende der Phase spiegelt die Schiefe der letzten Phase wider. Gültige Abmessungen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (job_performance) Gültige Statistiken: Durchschnitt, Maximum, Minimum, Prozent Einheit: Anzahl	

Metrik	Beschreibung	Kategorie
glue.driver.skewness.job	Metrikkategorie: job_performance Die Auftragsabweichung ist der gewichtete Durchschnitt der Abweichungen in den Auftragsphasen. Der gewichtete Durchschnitt misst Phasen, deren Ausführung länger dauert, eine größere Bedeutung bei. Auf diese Weise soll der Ausnahmefall vermieden werden, bei dem eine stark abweichende Phase im Vergleich zu anderen Phasen tatsächlich nur für eine sehr kurze Zeit ausgeführt wird (und ihre Abweichung daher für die gesamte Auftragsleistung nicht signifikant ist und es sich nicht lohnt, diese Abweichung zu beheben). Diese Metrik wird nach Abschluss jeder Phase aktualisiert, sodass der letzte Wert die tatsächliche Gesamtabweichung des Auftrags widerspiegelt. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL),	job_performance

Metrik	Beschreibung	Kategorie
	Typ (Gauge) und Observability Group (job_performance) Gültige Statistiken: Durchschnitt, Maximum, Minimum, Prozent Einheit: Anzahl	
glue.succeed.ALL	Metrikkategorie: Fehler Gesamtzahl der erfolgreichen Auftragsausführungen, für ein vollständiges Bild der Fehlerkategorien Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Anzahl) und ObservabilityGroup (Fehler) Gültige Statistiken: SUM Einheit: Anzahl	error

Metrik	Beschreibung	Kategorie
glue.error.ALL	Metrikkategorie: Fehler Gesamtzahl der Fehler bei Auftragsausführungen, für ein vollständiges Bild der Fehlerkategorien Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Anzahl) und ObservabilityGroup (Fehler) Gültige Statistiken: SUM Einheit: Anzahl	error

Metrik	Beschreibung	Kategorie
glue.error.[Fehlerkategorie]	Metrikkategorie: Fehler Dabei handelt es sich um eine Reihe von Metriken, die nur aktualisiert werden, wenn eine Auftragsausführung fehlschlägt. Die Fehlerkategorisierung hilft bei der Analyse und dem Debugging. Wenn eine Auftragsausführung fehlschlägt, wird der Fehler, der dazu geführt hat, kategorisiert und die entsprechende Metrik für die Fehlerkategorie wird auf 1 gesetzt. Dies hilft bei der Durchführung von Fehleranalysen im Laufe der Zeit sowie bei der Fehleranalyse für alle Jobs, um die häufigsten Fehlerkategorien zu identifizieren und mit deren Behebung zu beginnen. AWS Glue hat 28 Fehlerkategorien, darunter die Fehlerkategorien OUT_OF_MEMORY (Treiber und Executor), PERMISSION, SYNTAX und THROTTLING. Weitere Fehlerkategorien sind unter anderem COMPILATION, LAUNCH und TIMEOUT. Gültige Dimensionen: (der Name der JobName AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ	error

Metrik	Beschreibung	Kategorie
	(Anzahl) und Observabi lityGroup (Fehler) Gültige Statistiken: SUM Einheit: Anzahl	
glue.driver.workerUtilization	Metrikkategorie: resource_ utilization Der Prozentsatz der zugewiesenen Arbeitskräfte, die tatsächlich eingesetzt werden. Wenn nicht gut, kann Auto Scaling helfen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und Observabi lityGroup (resource_utilization) Gültige Statistiken: Durchschn itt, Maximum, Minimum, Prozent Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.heap.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Heap-Speicher des Treibers während der Auftragsausführung. Dies hilft, Trends bei der Speichernutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.heap.used.percentage	Metrikkategorie: resource_utilization Der belegte Heap-Speicher des Treibers während der Auftragsausführung (in %). Dies hilft, Trends bei der Speichernutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.non-heap.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Non-Heap-Speicher des Treibers während der Auftragsausführung. Dies hilft, Trends bei der Speichernutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.non-heap .used.percentage	Metrikkategorie: resource_ utilization Der belegte Non-Heap-Speicher des Treibers während der Auftragsa usführung (in %). Dies hilft, Trends bei der Speichern utzung zu verstehen, insbesondere im Zeitverla uf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und Observabi lityGroup (resource_utilization) Gültige Statistiken: Durchschn itt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.total.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Gesamtspeicher des Treibers während der Auftragsausführung. Dies hilft, Trends bei der Speichernutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.memory.total.used.percentage	Metrikkategorie: resource_utilization Der belegte Gesamtspeicher des Treibers während der Auftragsausführung (in %). Dies hilft, Trends bei der Speichernutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit Speicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.memory.heap.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Heap-Speicher der Executoren. ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization
glue.ALL.memory.heap.used.percentage	Metrikkategorie: resource_utilization Der belegte Heap-Speicher der Executoren (in %). ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.memory.non-heap.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Non-Heap-Speicher der Executoren. ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.memory.non-heap.used.percentage	Metrikkategorie: resource_utilization Der belegte Non-Heap-Speicher der Executoren (in %). ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.memory.total.[verfügbar | belegt]	Metrikkategorie: resource_utilization Der verfügbare/belegte Gesamtspeicher der Executoren. ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Byte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.memory.total.used.percentage	Metrikkategorie: resource_utilization Der belegte Gesamtspeicher der Executoren (in %). ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.disk.[available_GB | used_GB]	Metrikkategorie: resource_utilization Der verfügbare/belegte Festplattenspeicher des Treibers während der Auftragsausführung. Dies hilft, Trends bei der Festplatt ennutzung zu verstehen, insbesondere im Zeitverla uf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit zu wenig Festplattenspeicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und Observabi lityGroup (resource_utilization) Gültige Statistiken: Durchschn itt Einheit: Gigabyte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.disk.used.percentage]	Metrikkategorie: resource_utilization Der verfügbare/belegte Festplattenspeicher des Treibers während der Auftragsausführung. Dies hilft, Trends bei der Festplattennutzung zu verstehen, insbesondere im Zeitverlauf, wodurch potenzielle Ausfälle vermieden werden. Außerdem können Ausfälle im Zusammenhang mit zu wenig Festplattenspeicher behoben werden. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.disk.[available_GB | used_GB]	Metrikkategorie: resource_utilization Der verfügbare/belegte Festplattenspeicher der Executoren. ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Gigabyte	resource_utilization

Metrik	Beschreibung	Kategorie
glue.ALL.disk.used.percentage	Metrikkategorie: resource_utilization Der Festplattenspeicher (%) der Executorenavailable/used/used. ALL bedeutet alle Executoren. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Typ (Gauge) und ObservabilityGroup (resource_utilization) Gültige Statistiken: Durchschnitt Einheit: Prozentsatz	resource_utilization

Metrik	Beschreibung	Kategorie
glue.driver.bytesRead	Metrikkategorie: Durchsatz Die Anzahl der in dieser Auftragsausführung pro Eingabequelle sowie für ALL-Quellen gelesenen Byte. Dies hilft, das Datenvolumen und seine Veränderungen im Laufe der Zeit besser zu verstehen und Probleme wie Datenabweichungen zu lösen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Type (Gauge), ObservabilityGroup (resource_utilization) und Source (Quelldatenposition) Gültige Statistiken: Durchschnitt Einheit: Byte	Durchsatz

Metrik	Beschreibung	Kategorie
glue.driver.[recordsRead | filesRead]	Metrikkategorie: Durchsatz Die Anzahl der in dieser Auftragsausführung pro Eingabequelle sowie für ALL-Quellen gelesenen Datensätze/Dateien. Dies hilft, das Datenvolumen und seine Veränderungen im Laufe der Zeit besser zu verstehen und Probleme wie Datenabweichungen zu lösen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Type (Gauge), ObservabilityGroup (resource_utilization) und Source (Quelldatenposition) Gültige Statistiken: Durchschnitt Einheit: Anzahl	Durchsatz

Metrik	Beschreibung	Kategorie
glue.driver.partitionsRead	Metrikkategorie: Durchsatz Die Anzahl der in dieser Auftragsausführung pro Amazon-S3-Eingabequelle sowie für ALL-Quellen gelesenen Partitionen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Type (Gauge), ObservabilityGroup (resource_utilization) und Source (Quelldatenposition) Gültige Statistiken: Durchschnitt Einheit: Anzahl	Durchsatz

Metrik	Beschreibung	Kategorie
glue.driver.bytesWritten	Metrikkategorie: Durchsatz Die Anzahl der in dieser Auftragsausführung pro Ausgabe-Sink sowie für ALL-Sinks geschriebenen Byte. Dies hilft, das Datenvolumen und seine Entwicklung im Laufe der Zeit besser zu verstehen und Probleme wie Verarbeitungsabweichungen zu lösen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Type (Gauge), ObservabilityGroup (resource_utilization) und Sink (Speicherort der Senkendaten) Gültige Statistiken: Durchschnitt Einheit: Byte	Durchsatz

Metrik	Beschreibung	Kategorie
glue.driver.[recordsWritten | filesWritten]	Metrikkategorie: Durchsatz Die Anzahl der in dieser Auftragsausführung pro Ausgabe-Sink sowie für ALL-Sinks geschriebenen Datensätze/Dateien. Dies hilft, das Datenvolumen und seine Entwicklung im Laufe der Zeit besser zu verstehen und Probleme wie Verarbeitungsabweichungen zu lösen. Gültige Dimensionen: JobName (der Name der AWS Glue Job), JobRunId (die JobRun ID oder ALL), Type (Gauge), ObservabilityGroup (resource_utilization) und Sink (Speicherort der Senkendaten) Gültige Statistiken: Durchschnitt Einheit: Anzahl	Durchsatz

Fehlerkategorien

Fehlerkategorien	Beschreibung
COMPILATION_ERROR	Bei der Kompilierung von Scala-Code treten Fehler auf.

Fehlerkategorien	Beschreibung
CONNECTION_ERROR	Beim Herstellen einer Verbindung zu einem service/remote host/database Dienst usw. treten Fehler auf.
DISK_NO_SPACE_ERROR	Wenn auf der Festplatte des Treibers/ Executors kein Speicherplatz mehr vorhanden ist, treten Fehler auf.
OUT_OF_MEMORY_ERROR	Wenn auf dem Speicher des Treibers/ Executors kein Speicherplatz mehr vorhanden ist, treten Fehler auf.
IMPORT_ERROR	Beim Import von Abhängigkeiten treten Fehler auf.
INVALID_ARGUMENT_ERROR	Fehler treten auf, wenn die Eingabeargumente ungültig/illegal sind.
PERMISSION_ERROR	Wenn die Genehmigung für Services, Daten usw. fehlt, treten Fehler auf.
RESOURCE_NOT_FOUND_ERROR	Wenn Daten, Speicherorte usw. nicht existiere n, treten Fehler auf.
QUERY_ERROR	Bei der Ausführung von Spark-SQL-Abfragen treten Fehler auf.
SYNTAX_ERROR	Wenn das Skript einen Syntaxfehler enthält, treten Fehler auf.
THROTTLING_ERROR	Wenn die Beschränkung der Parallelität von Services erreicht oder die Beschränkung der Service Quotas überschritten wird, treten Fehler auf.

Fehlerkategorien	Beschreibung
DATA_LAKE_FRAMEWORK_ERROR	Fehler entstehen durch AWS Glue nativ unterstütztes Data-Lake-Framework wie Hudi, Iceberg usw.
UNSUPPORTED_OPERATION_ERROR	Wenn ein Vorgang ausgeführt wird, der nicht unterstützt wird, treten Fehler auf.
RESOURCES_ALREADY_EXISTS_ERROR	Wenn eine Ressource, die erstellt oder hinzugefügt werden soll, bereits vorhanden ist, treten Fehler auf.
GLUE_INTERNAL_SERVICE_ERROR	Fehler treten auf, wenn es eine AWS Glue internes Serviceproblem.
GLUE_OPERATION_TIMEOUT_ERROR	Fehler treten auf, wenn ein AWS Glue Der Vorgang ist Timeout.
GLUE_VALIDATION_ERROR	Fehler treten auf, wenn ein erforderlicher Wert nicht validiert werden konnte AWS Glue Arbeit.
GLUE_JOB_BOOKMARK_VERSION_MISMATCH_ERROR	Wenn derselbe Auftrag auf demselben Quell-Bucket ausgeführt und gleichzeitig auf dasselbe/unterschiedliche Ziel geschrieben wird (Gleichzeitigkeit >1), treten Fehler auf.
LAUNCH_ERROR	Fehler entstehen während der AWS Glue Phase der Jobeinführung.
DYNAMODB_ERROR	Generische Fehler entstehen durch Amazon DynamoDB den Service.
GLUE_ERROR	Generische Fehler entstehen durch AWS Glue Service.
LAKEFORMATION_ERROR	Generische Fehler entstehen durch AWS Lake Formation den Service.

Fehlerkategorien	Beschreibung
REDSHIFT_ERROR	Generische Fehler entstehen durch Amazon Redshift den Service.
S3_ERROR	Der Amazon–S3-Service ruft generische Fehler hervor.
SYSTEM_EXIT_ERROR	Generischer Fehler beim Beenden des Systems.
TIMEOUT_ERROR	Wenn der Auftrag aufgrund eines Timeouts fehlschlägt, treten generische Fehler auf.
UNCLASSIFIED_SPARK_ERROR	Spark ruft generische Fehler hervor.
UNCLASSIFIED_ERROR	Standard-Fehlerkategorie.

Einschränkungen

> **Note**
>
> `glueContext` muss initialisiert werden, um die Metriken zu veröffentlichen.

In der Quelldimension ist der Wert je nach Quelltyp entweder ein Amazon-S3-Pfad oder Tabellenname. Wenn es sich bei der Quelle um JDBC handelt und die Abfrageoption verwendet wird, wird die Abfragezeichenfolge außerdem in der Quelldimension festgelegt. Wenn der Wert länger als 500 Zeichen ist, wird er auf 500 Zeichen gekürzt. Für den Wert gelten folgende Einschränkungen:

- Nicht-ASCII-Zeichen werden entfernt.
- Wenn der Quellname kein ASCII-Zeichen enthält, wird er in <Nicht-ASCII-Eingabe> umgewandelt.

Einschränkungen und Überlegungen zu Durchsatzmetriken

- DataFrame und DataFrame based DynamicFrame (z. B. JDBC, Lesen von Parquet auf Amazon S3) werden unterstützt, RDD-basiert DynamicFrame (z. B. Lesen von CSV, JSON auf Amazon S3

usw.) wird jedoch nicht unterstützt. Technisch gesehen werden alle Lese- und Schreibvorgänge, die auf der Spark-Benutzeroberfläche sichtbar sind, unterstützt.

- Die `recordsRead`-Metrik wird ausgegeben, wenn es sich bei der Datenquelle um eine Katalogtabelle handelt und das Format JSON, CSV, Text oder Iceberg ist.
- Die Metriken `glue.driver.throughput.recordsWritten`, `glue.driver.throughput.bytesWritten` und `glue.driver.throughput.filesWritten` sind in JDBC- und Iceberg-Tabellen nicht verfügbar.
- Metriken können verzögert sein. Wenn der Job in etwa einer Minute abgeschlossen ist, gibt es unter Metrics möglicherweise keine Durchsatzmetriken. Amazon CloudWatch

Auftragsüberwachung und Debugging

Sie können Metriken sammeln über AWS Glue Jobs und visualisieren Sie sie auf der AWS Glue und CloudWatch Amazon-Konsolen zur Identifizierung und Behebung von Problemen. Profilierung Ihres AWS Glue Jobs erfordert die folgenden Schritte:

1. Metriken aktivieren:

   a. Aktivieren der Option Job metrics (Auftragsmetriken) in der Auftragsdefinition. Sie können die Profilerstellung aktivieren in AWS Glue Konsole oder als Parameter für den Job. Weitere Informationen finden Sie unter Definieren von Auftragseigenschaften für Spark-Aufträge oder Verwenden von Auftragsparametern in AWS Glue-Jobs.

   b. Aktivieren des AWS Glue Option für Messwerte zur Beobachtbarkeit in der Jobdefinition. Sie können Observability aktivieren in AWS Glue Konsole oder als Parameter für den Job. Weitere Informationen finden Sie unter Überwachen mit AWS Glue Beobachtbarkeitsmetriken.

2. Vergewissern Sie sich, dass das Auftragsskript einen `GlueContext` initialisiert. Beispiel: Der folgende Skriptausschnitt initialisiert einen `GlueContext` und zeigt, wo der profilierte Code im Skript platziert wird. Dieses allgemeine Format wird in den folgenden Debugging-Szenarien verwendet.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

...
...
code-to-profile
...
...


job.commit()
```

3. Führen Sie den Auftrag aus.
4. Metriken visualisieren:
   a. Visualisieren Sie Job-Metriken auf der AWS Glue Konsolen Sie und identifizieren Sie ungewöhnliche Messwerte für den Fahrer oder einen Testamentsvollstrecker.
   b. Überprüfen Sie die Messwerte zur Beobachtbarkeit auf der Seite zur Überwachung von Jobläufen, auf der Seite mit den Details zur Jobausführung oder bei Amazon CloudWatch. Weitere Informationen finden Sie unter Überwachen mit AWS Glue Beobachtbarkeitsmetriken.
5. Grenzen Sie die Ursache mithilfe der identifizierten Metrik ein.
6. Optional bestätigen Sie die Ursache mithilfe des Protokollstreams des identifizierten Treibers oder Auftrags-Executors.

Anwendungsfälle für AWS Glue Messwerte zur Beobachtbarkeit

- Debuggen von OOM-Ausnahmen und Auftragsabweichungen
- Debugging von anspruchsvolle Phasen und Straggler-Aufgaben

- Überwachen des Fortschritts mehrerer Aufträge
- Überwachung für die DPU-Kapazitätsplanung
- Verwenden AWS Glue Beobachtbarkeit zur Überwachung der Ressourcennutzung zur Kostensenkung

Debuggen von OOM-Ausnahmen und Auftragsabweichungen

Sie können Ausnahmen und Jobanomalien debuggen out-of-memory (OOM) in AWS Glue. In den folgenden Abschnitten werden Szenarien für das Debuggen von out-of-memory Ausnahmen des Apache Spark-Treibers oder eines Spark-Executors beschrieben.

- Debuggen einer OOM-Ausnahme für einen Treiber
- Debuggen einer OOM-Ausnahme eines Executors

Debuggen einer OOM-Ausnahme für einen Treiber

In diesem Szenario liest ein Spark-Auftrag eine große Anzahl von kleinen Dateien von Amazon Simple Storage Service (Amazon S3). Er wandelt die Dateien in das Apache-Parquet-Format um und schreibt sie in Amazon S3. Der Spark-Treiber verfügt nicht mehr über genügend Arbeitsspeicher. Die eingegebenen Amazon-S3-Daten umfassen mehr als 1 Million Dateien in verschiedenen Amazon-S3-Partitionen.

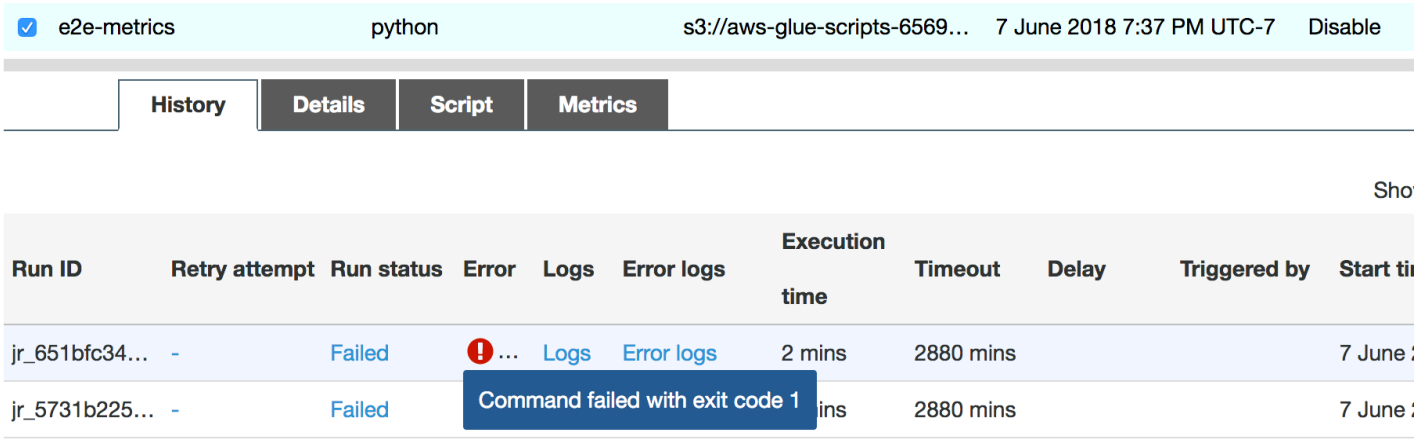
Der profilierte Code sieht wie folgt aus:

```
data = spark.read.format("json").option("inferSchema", False).load("s3://input_path")
data.write.format("parquet").save(output_path)
```

Visualisieren der profilierten Metriken auf der AWS Glue -Konsole

Das folgende Diagramm zeigt die Speicherauslastung als Prozentsatz für den Treiber und Executors. Diese Nutzung wird als ein Datenpunkt dargestellt, gemittelt über die Werte in der letzten Minute. Sie sehen im Speicherprofil des Auftrags, dass der Speicher für den Treiber die sichere Schwelle von 50 Prozent Auslastung schnell überschreitet. Zum anderen ist die durchschnittliche Speicherauslastung für alle Executors nach wie vor kleiner als 4 Prozent. Dies zeigt deutlich Anomalien bei der Treiberausführung in diesem Spark-Auftrag.

Die Ausführung des Jobs schlägt bald fehl, und der folgende Fehler wird auf der Registerkarte Verlauf auf der AWS Glue Konsole: Befehl mit Exitcode 1 fehlgeschlagen. Diese Fehlermeldung bedeutet, dass der Auftrag aufgrund eines systemischen Fehlers fehlgeschlagen ist. In diesem Fall steht dem Treiber nicht genügend Arbeitsspeicher zur Verfügung.



Klicken Sie auf der Konsole auf der Registerkarte Verlauf auf den Link Fehlerprotokolle, um zu bestätigen, dass Sie in den CloudWatch Protokollen den Treiber OOM gefunden haben. Suchen Sie in den Fehlerprotokollen des Auftrags nach "**`Error`**" (Fehler), um zu bestätigen, dass es tatsächlich eine OOM-Ausnahme war, aufgrund derer der Auftrag fehlgeschlagen ist:

```
# java.lang.OutOfMemoryError: Java heap space
```

```
# -XX:OnOutOfMemoryError="kill -9 %p"
# Executing /bin/sh -c "kill -9 12039"...
```

Wählen Sie auf der Registerkarte History (Verlauf) für den Auftrag die Option Logs (Protokolle). Zu Beginn des Jobs finden Sie in den CloudWatch Protokollen den folgenden Verlauf der Treiberausführung. Der Spark-Treiber versucht, alle Dateien in allen Verzeichnissen aufzulisten, erzeugt einen `InMemoryFileIndex` und startet eine Aufgabe pro Datei. Dies wiederum führt dazu, dass der Spark-Treiber eine große Datenmenge im Arbeitsspeicher halten muss, um den Status aller Aufgaben zu verfolgen. Er speichert die vollständige Liste einer großen Anzahl von Dateien für den In-Memory-Index im Cache, was zu einem OOM-Fehler für den Treiber führt.

Korrigieren Sie die Verarbeitung von mehreren Dateien unter Verwendung einer Gruppierung

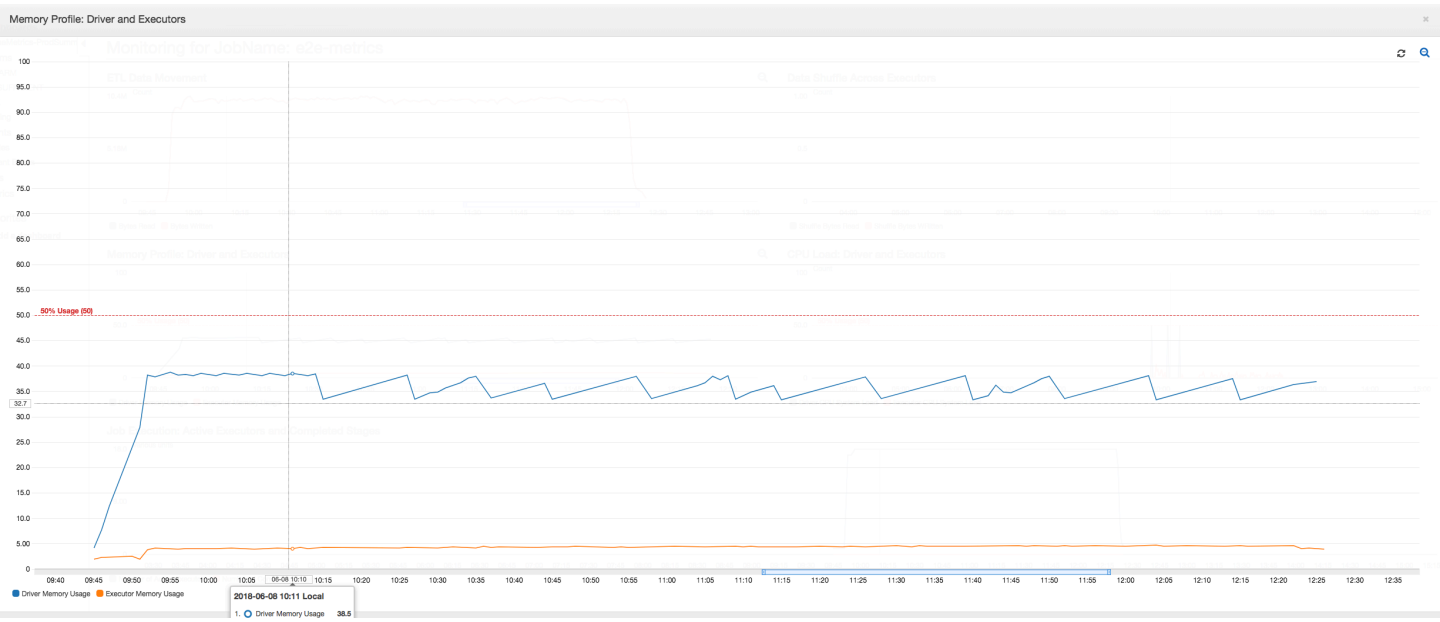
Sie können die Verarbeitung mehrerer Dateien korrigieren, indem Sie die Gruppierungsfunktion in verwenden AWS Glue. Die Gruppierung wird automatisch aktiviert, wenn Sie dynamische Frames verwenden und wenn das Eingabe-Dataset eine große Anzahl von Dateien enthält (mehr als 50.000). Die Gruppierung ermöglicht es Ihnen, mehrere Dateien zu einer Gruppe zusammenzufassen, und erlaubt es einer Aufgabe, die gesamte Gruppe statt einer einzelnen Datei zu bearbeiten. Infolgedessen speichert der Spark-Treiber deutlich weniger Status im Speicher, um weniger Aufgaben zu verfolgen. Weitere Informationen zum manuellen Aktivieren der Gruppierung für Ihre Datasets finden Sie unter Zusammenfassen von Eingabedateien in größeren Gruppen beim Lesen.

Um das Speicherprofil des zu überprüfen AWS Glue Job, profilieren Sie den folgenden Code mit aktivierter Gruppierung:

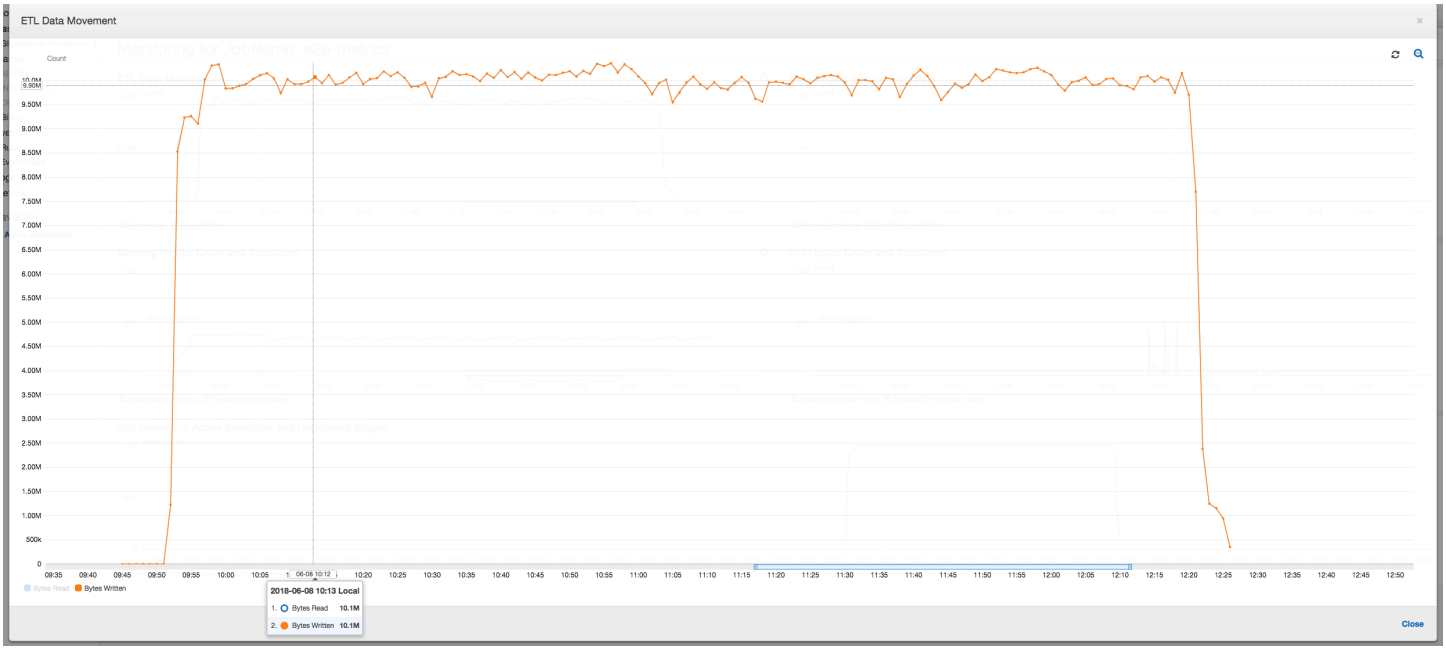
```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type
 = "s3", connection_options = {"path": output_path}, format = "parquet",
 transformation_ctx = "datasink")
```

Sie können das Speicherprofil und die ETL-Datenbewegung im AWS Glue Berufsprofil.

Der Treiber unterschreitet den Schwellenwert von 50 Prozent Speicherauslastung während der gesamten Dauer des AWS Glue Job. Die Executors streamen die Daten von Amazon S3, verarbeiten sie und schreiben sie in Amazon S3. Dadurch verbrauchen sie zu jedem Zeitpunkt weniger als 5 Prozent Speicherplatz.



Das folgende Datenbewegungsprofil zeigt die Gesamtzahl der Amazon-S3-Bytes, die beim Auftragsfortschritt in der letzten Minute von allen Executors gelesen und geschrieben wurden. Beide folgen einem ähnlichen Muster, da die Daten über alle Executors gestreamt werden. Der Auftrag verarbeitet alle eine Million Dateien in weniger als drei Stunden.



Debuggen einer OOM-Ausnahme eines Executors

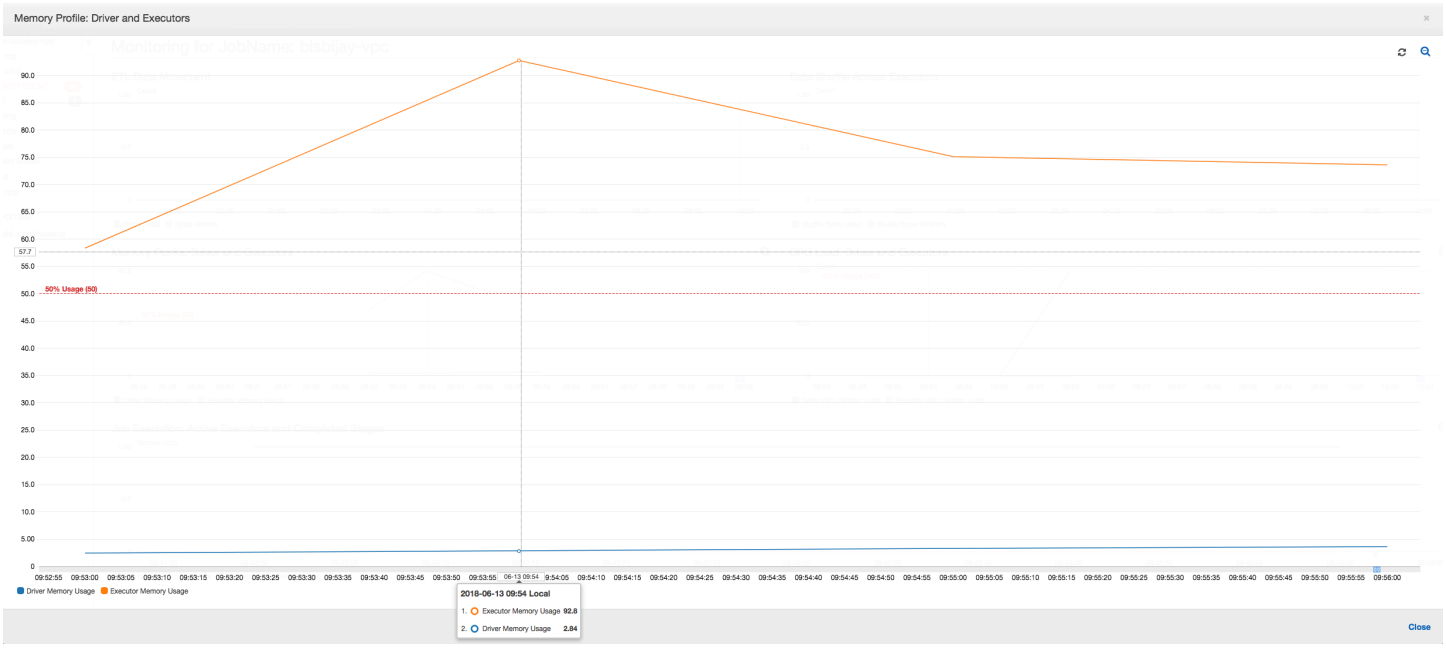
In diesem Szenario erfahren Sie, wie Sie OOM-Ausnahmen debuggen, die in Apache Spark Executors auftreten können. Der folgende Code verwendet den Spark MySQL-Reader, um eine große Tabelle mit etwa 34 Millionen Zeilen in einen Spark DataFrame einzulesen. Anschließend

schreibt es ihn im Parquet-Format in Amazon S3. Sie können die Verbindungseigenschaften angeben und die Standardkonfigurationen von Spark verwenden, um die Tabelle zu lesen.

```
val connectionProperties = new Properties()
connectionProperties.put("user", user)
connectionProperties.put("password", password)
connectionProperties.put("Driver", "com.mysql.jdbc.Driver")
val sparkSession = glueContext.sparkSession
val dfSpark = sparkSession.read.jdbc(url, tableName, connectionProperties)
dfSpark.write.format("parquet").save(output_path)
```

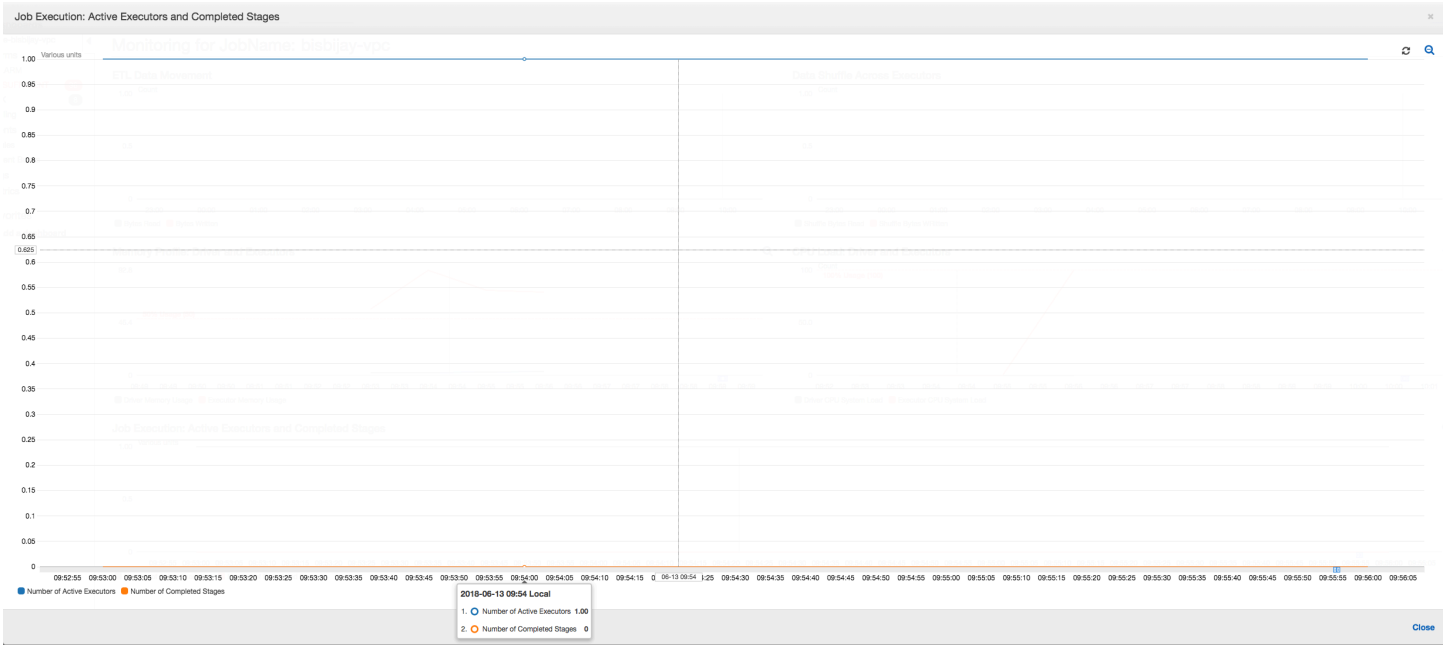
Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole

Wenn die Steigung des Speicherauslastungsdiagramms positiv ist und 50 Prozent überschreitet und der Auftrag fehlschlägt, bevor die nächste Metrik ausgegeben wird, wird dies wahrscheinlich durch aufgebrauchten Speicher verursacht. Das folgende Diagramm zeigt, dass innerhalb einer Minute der Ausführung die durchschnittliche Speicherauslastung in allen Executors schnell über 50 Prozent steigt. Die Nutzung erreicht bis zu 92 Prozent und der Container, der den Executor ausführt, wird von Apache Hadoop YARN beendet.

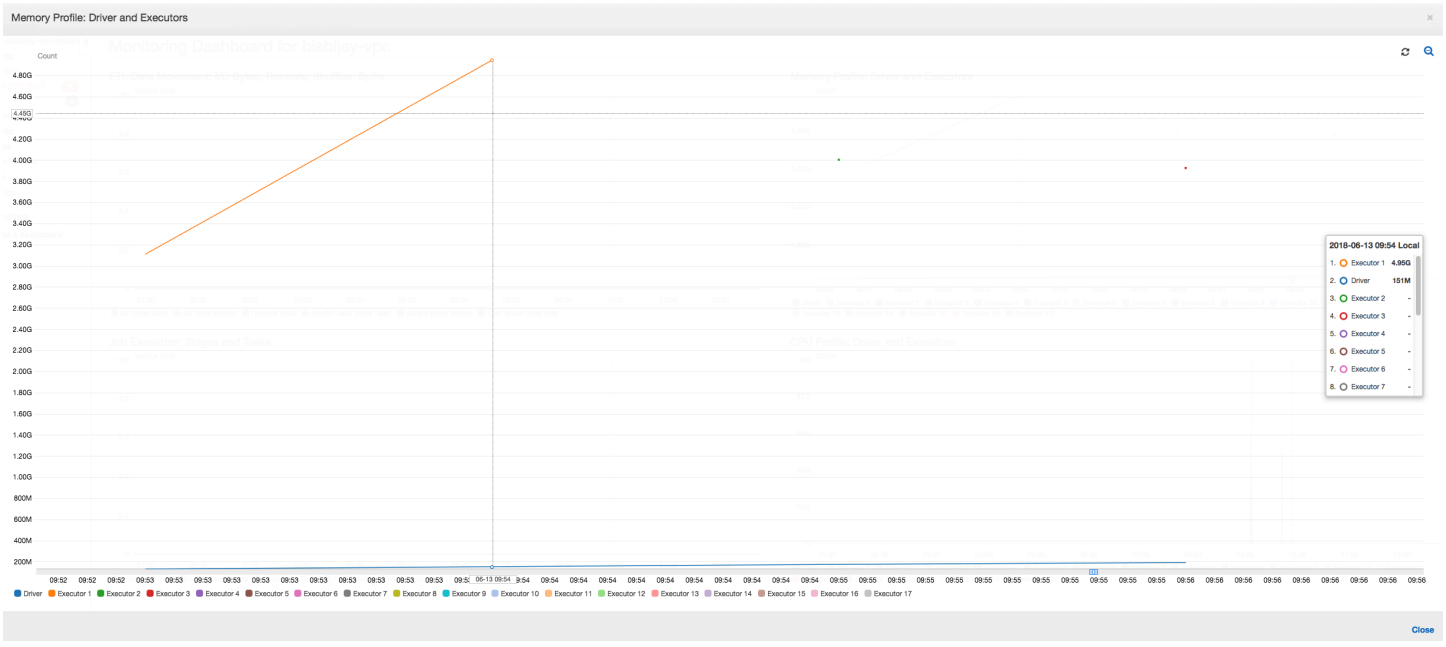


Wie die folgende Grafik zeigt, wird immer ein einzelner Executor ausgeführt, bis der Auftrag fehlschlägt. Dies liegt daran, dass ein neuer Executor gestartet wird, um den beendeten Executor zu ersetzen. Die Lesevorgänge der JDBC-Datenquelle sind standardmäßig nicht parallelisiert, da dies

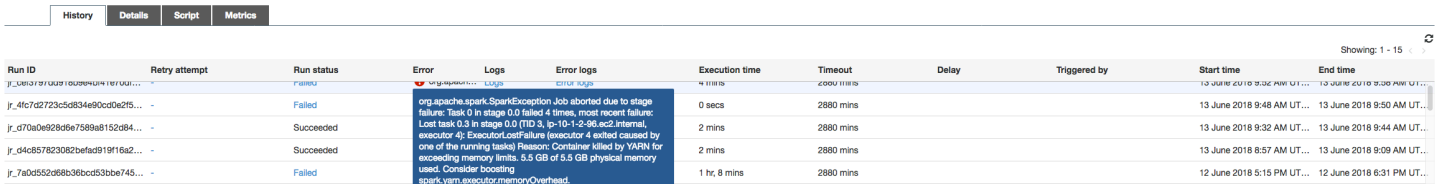
eine Partitionierung der Tabelle für eine Spalte und das Öffnen mehrerer Verbindungen erforderlich machen würde. Dies bewirkt, dass nur ein Executor die vollständige Tabelle sequenziell liest.



Wie die folgende Grafik zeigt, versucht Spark viermal, eine neue Aufgabe zu starten, bevor der Auftrag fehlschlägt. Sie sehen das Speicherprofil von drei Executors. Jeder Executor verbraucht schnell seinen gesamten Arbeitsspeicher. Der vierte Executor hat nicht mehr genügend Speicherplatz, und der Auftrag schlägt fehl. Aus diesem Grund wird die Metrik nicht sofort gemeldet.



Sie können dies anhand der Fehlerzeichenfolge auf der AWS Glue Konsole, dass der Job aufgrund von OOM-Ausnahmen fehlgeschlagen ist, wie in der folgenden Abbildung dargestellt.



Jobausgabeprotokolle: Weitere Informationen zur Bestätigung Ihrer Feststellung einer Executor-OOM-Ausnahme finden Sie in den CloudWatch Protokollen. Wenn Sie nach **`Error`** suchen, finden Sie zuerst die vier Executors, die etwa in denselben Zeitfenstern beendet wurden, wie auf dem Metriken-Dashboard gezeigt. Alle werden von YARN beendet, wenn sie ihre Speicherlimits überschreiten.

Executor 1

```
18/06/13 16:54:29 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 ERROR YarnClusterScheduler: Lost executor 1 on
 ip-10-1-2-175.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN TaskSetManager: Lost task 0.0 in stage 0.0 (TID 0,
 ip-10-1-2-175.ec2.internal, executor 1): ExecutorLostFailure (executor 1
 exited caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Executor 2

```
18/06/13 16:55:35 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 ERROR YarnClusterScheduler: Lost executor 2 on
 ip-10-1-2-16.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

```
18/06/13 16:55:35 WARN TaskSetManager: Lost task 0.1 in stage 0.0 (TID 1,
 ip-10-1-2-16.ec2.internal, executor 2): ExecutorLostFailure (executor 2 exited
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Executor 3

```
18/06/13 16:56:37 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 ERROR YarnClusterScheduler: Lost executor 3 on
 ip-10-1-2-189.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN TaskSetManager: Lost task 0.2 in stage 0.0 (TID 2,
 ip-10-1-2-189.ec2.internal, executor 3): ExecutorLostFailure (executor 3
 exited caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Executor 4

```
18/06/13 16:57:18 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 ERROR YarnClusterScheduler: Lost executor 4 on
 ip-10-1-2-96.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN TaskSetManager: Lost task 0.3 in stage 0.0 (TID 3,
 ip-10-1-2-96.ec2.internal, executor 4): ExecutorLostFailure (executor 4 exited
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

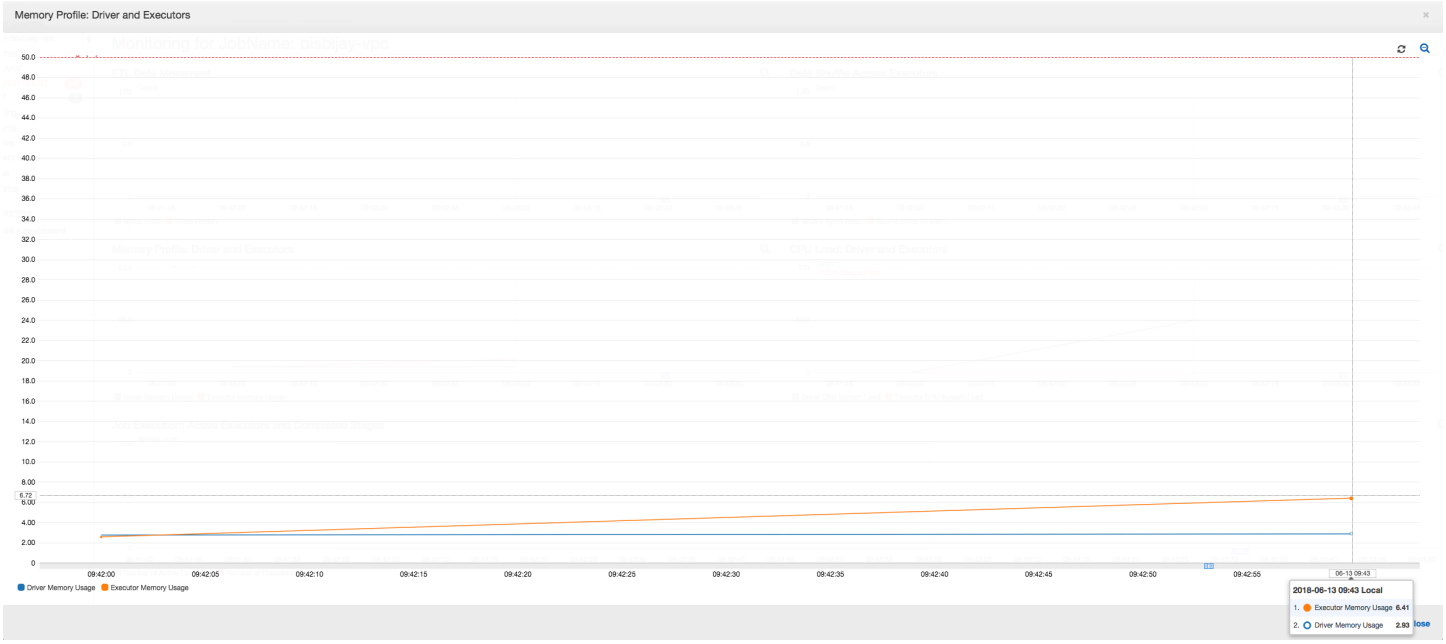
Korrigieren Sie die Einstellung für die Abrufgröße mit AWS Glue dynamische Frames

Der Executor hatte beim Lesen der JDBC-Tabelle keinen Speicher mehr zur Verfügung, da die Standardkonfiguration für die Spark JDBC-Abrufgröße Null ist. Dies bedeutet, dass der JDBC-Treiber auf dem Spark Executor versucht, die 34 Millionen Zeilen aus der Datenbank zusammen abzurufen und zwischenzuspeichern, obwohl Spark die Zeilen einzeln durchläuft. Mit Spark können Sie dieses Szenario zu vermeiden, indem Sie den Parameter für die Abrufgröße auf einen Standardwert ungleich Null setzen.

Sie können dieses Problem auch beheben, indem Sie AWS Glue stattdessen dynamische Frames. Dynamische Frames verwenden standardmäßig eine Abrufgröße von 1.000 Zeilen. Dieser Wert ist in der Regel ausreichend. Dies bewirkt, dass der Executor nicht mehr als 7 Prozent des gesamten Speichers verbraucht. Das Tool AWS Glue Der Job ist in weniger als zwei Minuten mit nur einem Executor abgeschlossen. Während der Verwendung AWS Glue Dynamische Frames sind der empfohlene Ansatz. Es ist auch möglich, die Abrufgröße mithilfe der Apache `fetchsize` Spark-Eigenschaft festzulegen. Weitere Informationen finden Sie im Spark SQL DataFrames and Datasets Guide.

```
val (url, database, tableName) = {
 ("jdbc_url", "db_name", "table_name")
 }
val source = glueContext.getSource(format, sourceJson)
val df = source.getDynamicFrame
glueContext.write_dynamic_frame.from_options(frame = df, connection_type = "s3",
 connection_options = {"path": output_path}, format = "parquet", transformation_ctx =
 "datasink")
```

Normale profilierte Metriken: Der Executor-Speicher mit AWS Glue dynamische Frames überschreiten niemals den sicheren Schwellenwert, wie in der folgenden Abbildung dargestellt. Er streamt die Zeilen aus der Datenbank und speichert jeweils nur 1.000 Zeilen im JDBC-Treiber. Es tritt keine Ausnahme aufgrund von Speichermangel auf.



Debugging von anspruchsvolle Phasen und Straggler-Aufgaben

Sie können Folgendes verwenden … AWS Glue Erstellung von Stellenprofilen zur Identifizierung anspruchsvoller Phasen und nachzügiger Aufgaben in Ihren ETL-Jobs (Extrahieren, Transformieren und Laden). Eine Nachzügler-Aufgabe dauert viel länger als die übrigen Aufgaben in einer Phase eines AWS Glue Arbeit. Dies bewirkt, dass die Phase länger dauert, was auch die gesamte Ausführungszeit des Auftrags verzögert.

Zusammenführung von kleinen Eingabedateien zu größeren Ausgabedateien

Eine Straggler-Aufgabe kann auftreten, wenn eine uneinheitliche Verteilung der Arbeit auf die verschiedenen Aufgaben gibt, oder wenn eine Datenverzerrung dazu führt, dass eine Aufgabe mehr Daten verarbeiten muss.

Sie können den folgenden Code profilieren (ein gemeinsames Muster in Apache Spark), um eine große Anzahl von kleinen Dateien zu größeren Ausgabedateien zusammenzuführen. In diesem Beispiel besteht das Eingabe-Dataset aus 32 GB mit Gzip komprimierten JSON-Dateien. Das Ausgabe-Dataset umfasst ca. 190 GB nicht komprimierte JSON-Dateien.

Der profilierte Code sieht wie folgt aus:

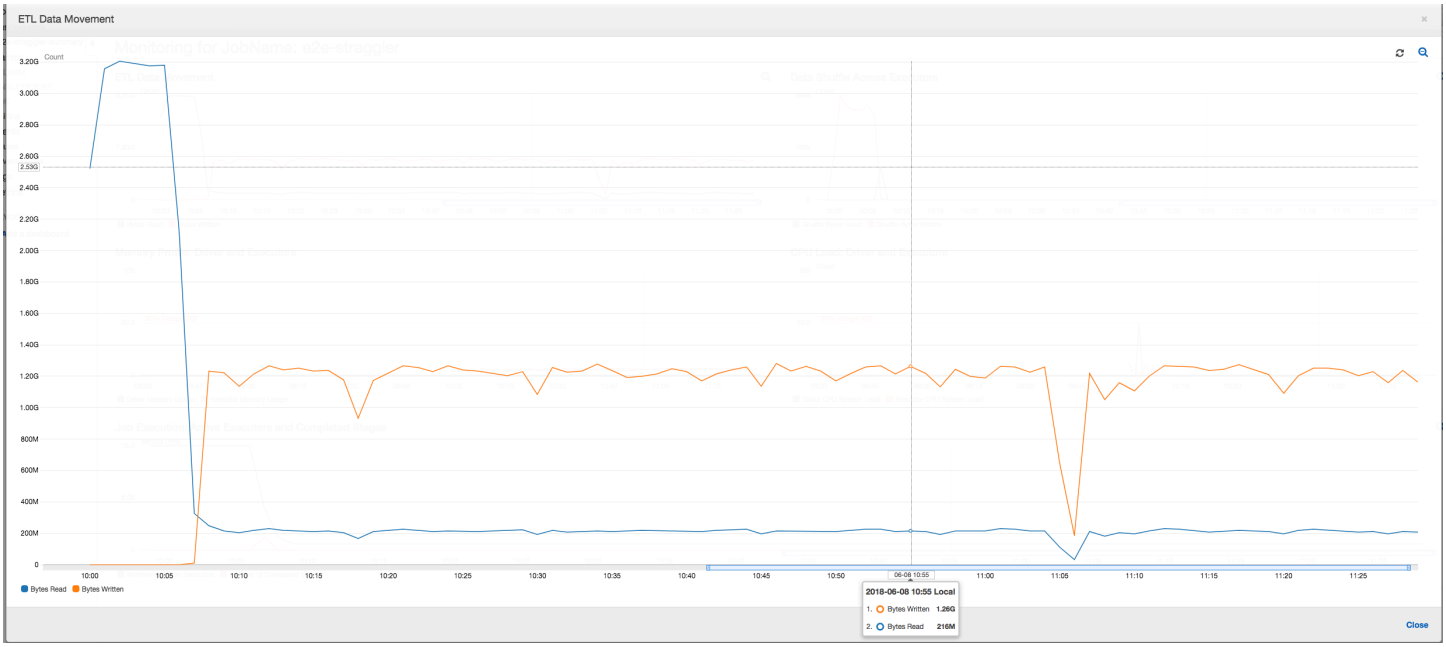
```
datasource0 = spark.read.format("json").load("s3://input_path")
df = datasource0.coalesce(1)
df.write.format("json").save(output_path)
```

Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole

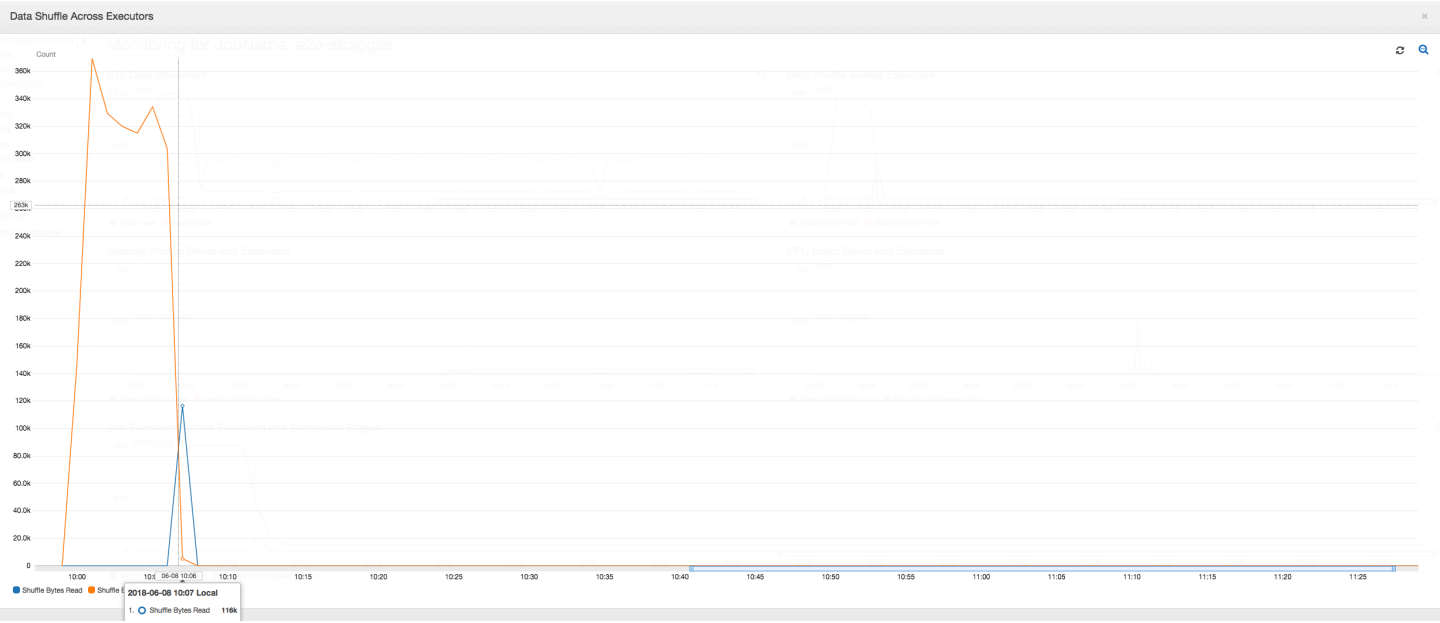
Sie können Ihren Auftrags profilieren, um vier verschiedene Metrik-Gruppen zu überprüfen:

- ETL-Datenbewegung
- Datenmischung über Executors hinweg
- Auftragsausführung
- Arbeitsspeicherprofil

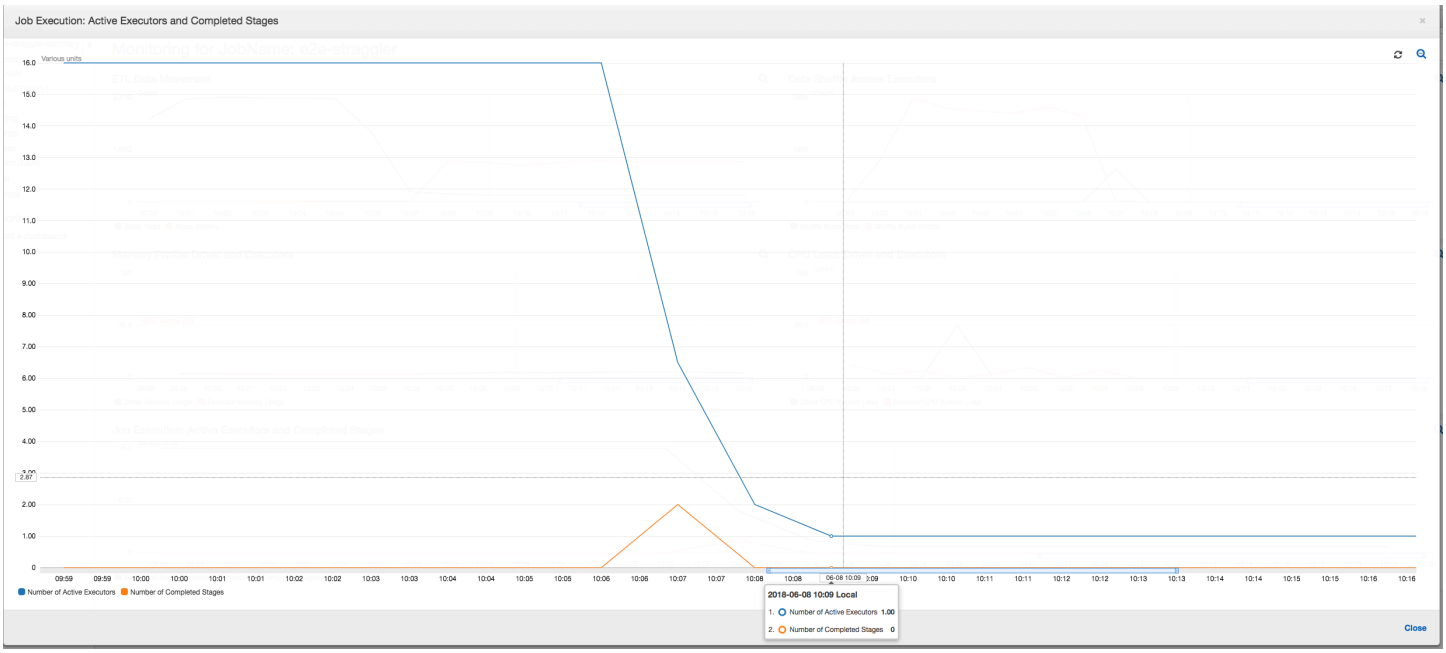
ETL-Datenbewegung: Im Profil ETL Data Movement (ETL-Datenbewegung) werden die Bytes von allen Executors in der ersten Phase, die innerhalb der ersten sechs Minuten abgeschlossen wird, relativ schnell gelesen. Die gesamte Auftragsausführung dauert jedoch etwa eine Stunde, vor allem durch die Schreibvorgänge für die Daten.



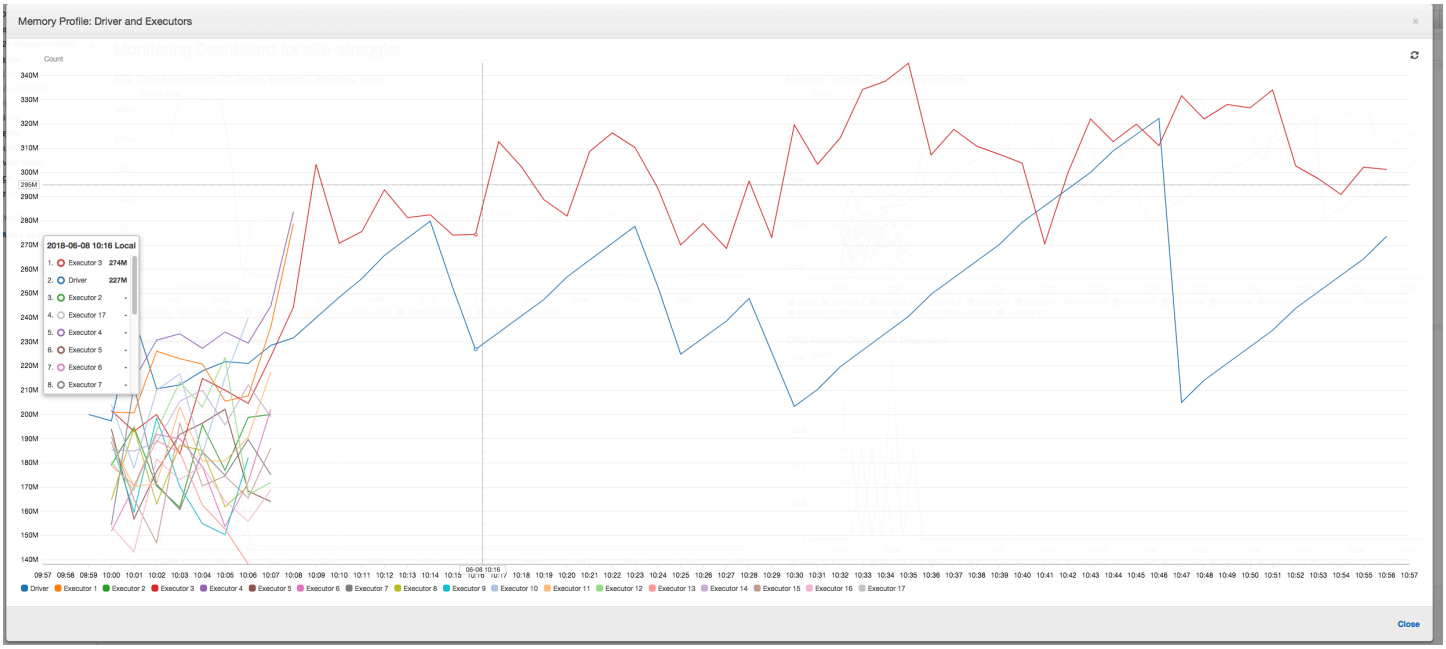
Datenmischung über Executors hinweg: Die Anzahl der gelesenen und geschriebenen Bytes während des Mischens zeigt auch eine hohe Zahl, bevor Phase 2 endet, wie durch die Metriken Job Execution (Auftragsausführung) und Data Shuffle (Datenmischung) gezeigt. Nachdem die Daten aus allen Executors gemischt wurden, werden die Lese- und Schreiboperationen nur noch von Executor Nummer 3 ausgeführt.



Auftragsausführung: Wie im folgenden Graphen gezeigt, sind alle anderen Executors im Leerlauf und werden schließlich zum Zeitpunkt 10:09 aufgegeben. Zu diesem Zeitpunkt verringert sich die Gesamtanzahl der Executors auf nur einen. Dies zeigt deutlich, dass der Executor Nummer 3 aus der Straggler-Aufgabe besteht, die die längste Ausführungszeit benötigt und zum größten Teil zur Auftragsausführungszeit beiträgt.



Speicherprofil: Nachdem den ersten beiden Phasen verbraucht nur noch Executor Nummer 3 aktiv Arbeitsspeicher zur Verarbeitung der Daten. Die restlichen Executors sind einfach im Leerlauf oder wurden kurz nach Abschluss der ersten beiden Phasen aufgegeben.



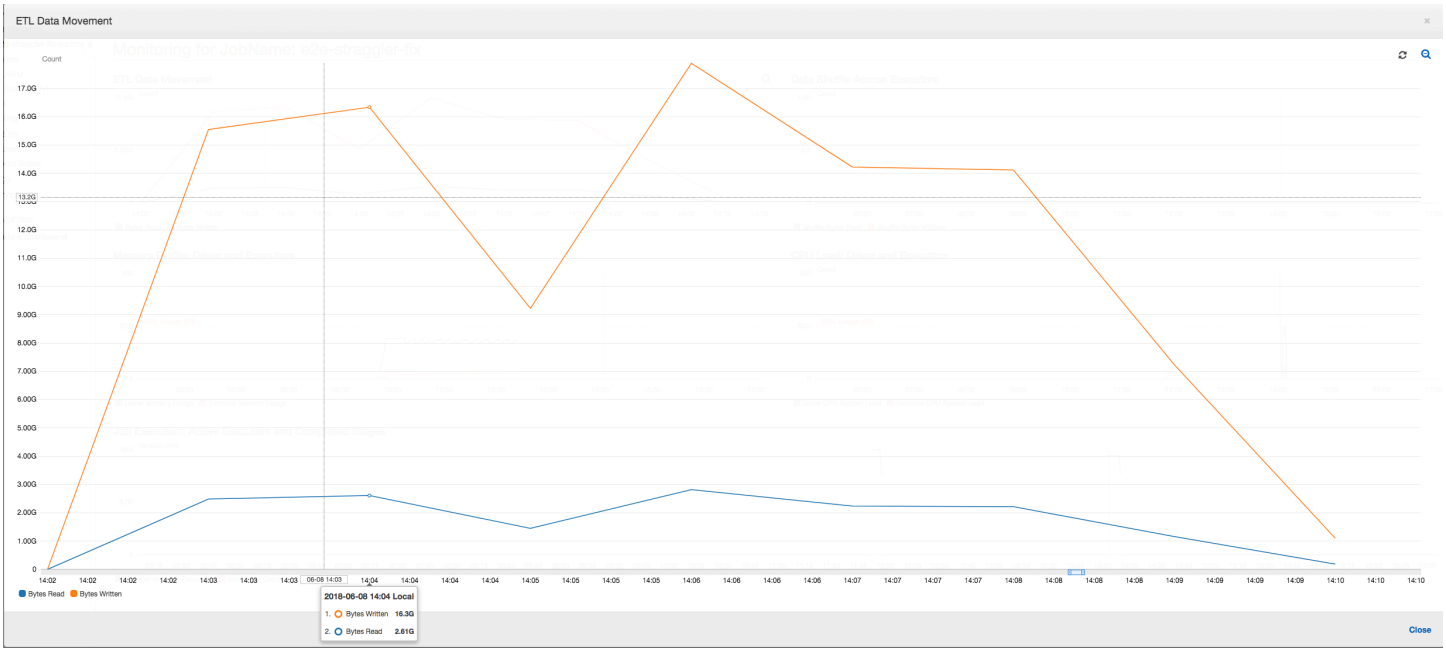
Beheben von Straggling-Executors unter Verwendung der Gruppierung

Sie können vermeiden, dass die Testamentsvollstrecker zu lange suchen, indem Sie die Gruppierungsfunktion in verwenden AWS Glue. Verwenden Sie Gruppierung, um die Daten gleichmäßig auf alle Executoren zu verteilen und Dateien mithilfe aller verfügbaren Executoren auf dem Cluster zu größeren Dateien zusammenzuführen. Weitere Informationen finden Sie unter Zusammenfassen von Eingabedateien in größeren Gruppen beim Lesen.

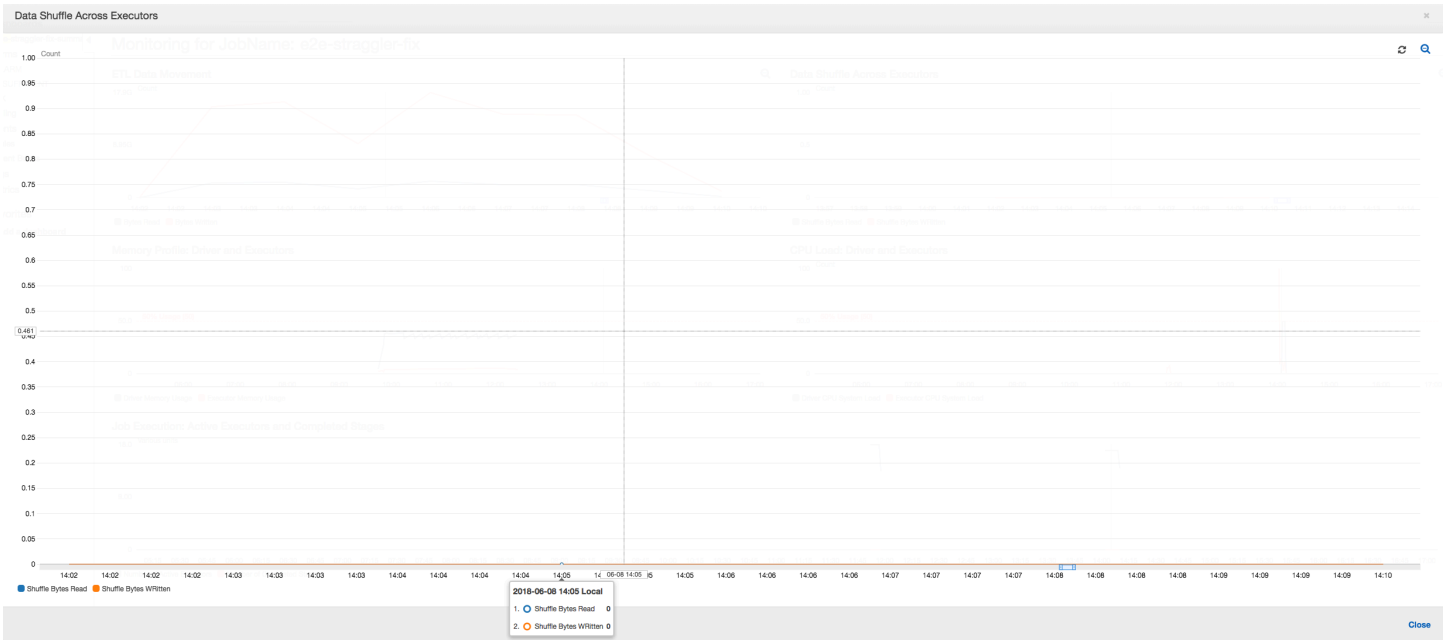
Um die ETL-Datenbewegungen im zu überprüfen AWS Glue Job, profilieren Sie den folgenden Code mit aktivierter Gruppierung:

```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type =
 "s3", connection_options = {"path": output_path}, format = "json", transformation_ctx
 = "datasink4")
```

ETL-Datenbewegung: Die Datenschreibvorgänge werden nun parallel zu den Datenlesevorgängen während der gesamten Ausführungszeit des Auftrags gestreamt. Dies bewirkt, dass der Auftrag innerhalb von acht Minuten abgeschlossen wird, viel schneller als zuvor.

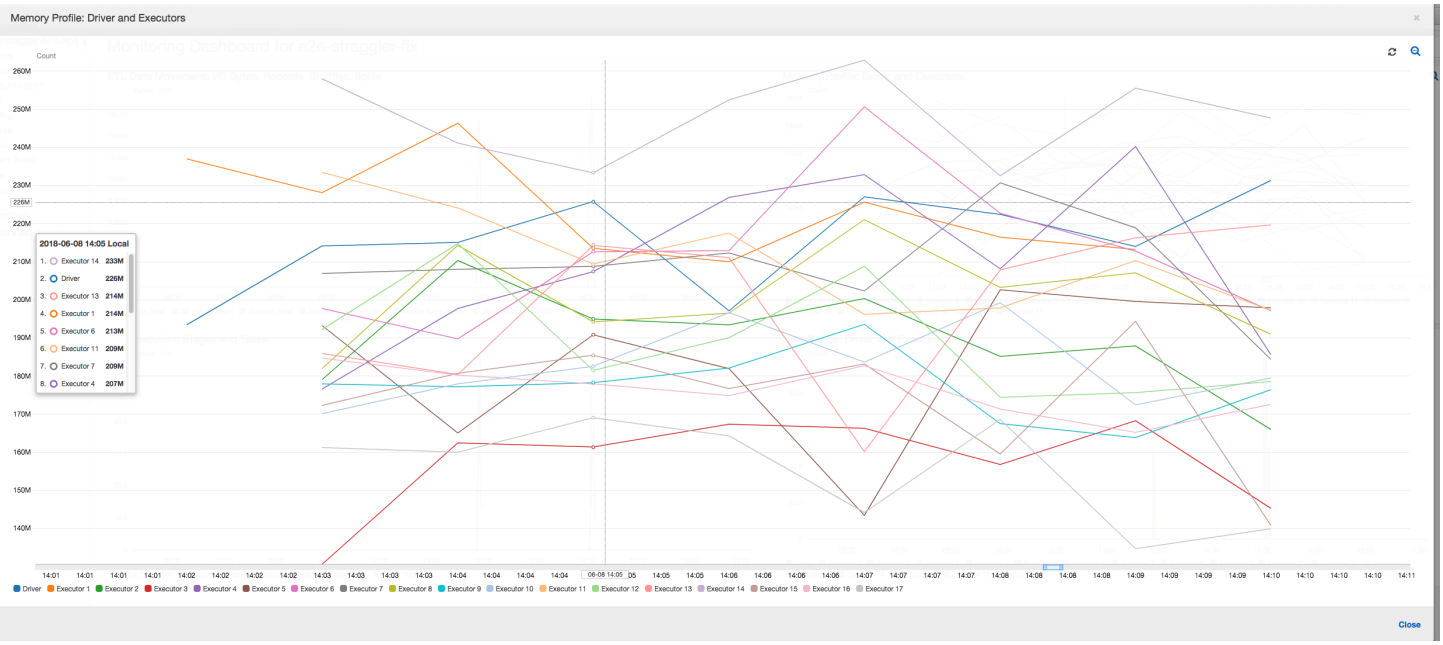


Datenmischung über Executors hinweg: Da die Eingabedateien während des Lesens über die Gruppierungsfeature zusammengeführt werden, gibt es nach dem Lesen der Daten keine kostspielige Datenmischung.



Auftragsausführung: Die Auftragsausführungsmetriken zeigen, dass die Gesamtanzahl der aktiven Executors, die ausgeführt werden und Daten verarbeiten, relativ konstant bleibt. Es gibt keine einzelnen Straggler in dem Auftrag. Alle Executors sind aktiv und werden nicht aufgegeben, bis zum der Auftrag abgeschlossen ist. Da es keine zwischenzeitliche Datenmischung zwischen den Executors gibt, gibt es nur eine einzige Phase in dem Auftrag.

Speicherprofil: Die Metriken zeigen die aktive Speicherbelegung für alle Executors – dies bestätigt, dass Aktivitäten auf allen Executors vorliegen. Da die Daten parallel gestreamt und ausgegeben werden, ist der gesamte Speicherbedarf aller Executors etwa gleich groß und liegt weit unter der sicheren Schwelle für alle Executors.



Überwachen des Fortschritts mehrerer Aufträge

Sie können mehrere AWS Glue Jobs zusammen profilieren und den Datenfluss zwischen ihnen überwachen. Dies ist ein gängiges Workflow-Muster und erfordert die Überwachung des Fortschritts

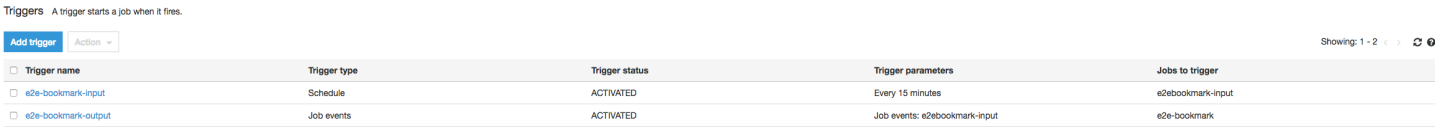
einzelner Aufträge, des Datenverarbeitungsrückstands, der Datenwiederaufbereitung und der Auftragslesezeichen.

Themen

- Profilierter Code
- Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole
- Korrektur der Dateiverarbeitung

Profilierter Code

In diesem Workflow haben Sie zwei Aufträge: einen Eingangsauftrag und einen Ausgangsauftrag. Die Ausführung des Eingangsauftrags ist für alle 30 Minuten eingeplant, wofür ein regelmäßiger Auslöser verwendet wird. Die Ausführung des Ausgangsauftrags ist nach jeder erfolgreichen Ausführung des Eingangsauftrags eingeplant. Diese geplanten Aufträge werden unter Verwendung von Auftragsauslöser kontrolliert.



Eingangsauftrag: Dieser Auftrag liest Daten von einem Amazon Simple Storage Service (Amazon S3)-Speicherort, transformiert sie mit `ApplyMapping` und schreibt sie an einen Amazon-S3-Staging--Speicherort. Der folgende Code ist profilierter Code für den Eingangsauftrag:

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": ["s3://input_path"],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": staging_path, "compression":
 "gzip"}, format = "json")
```

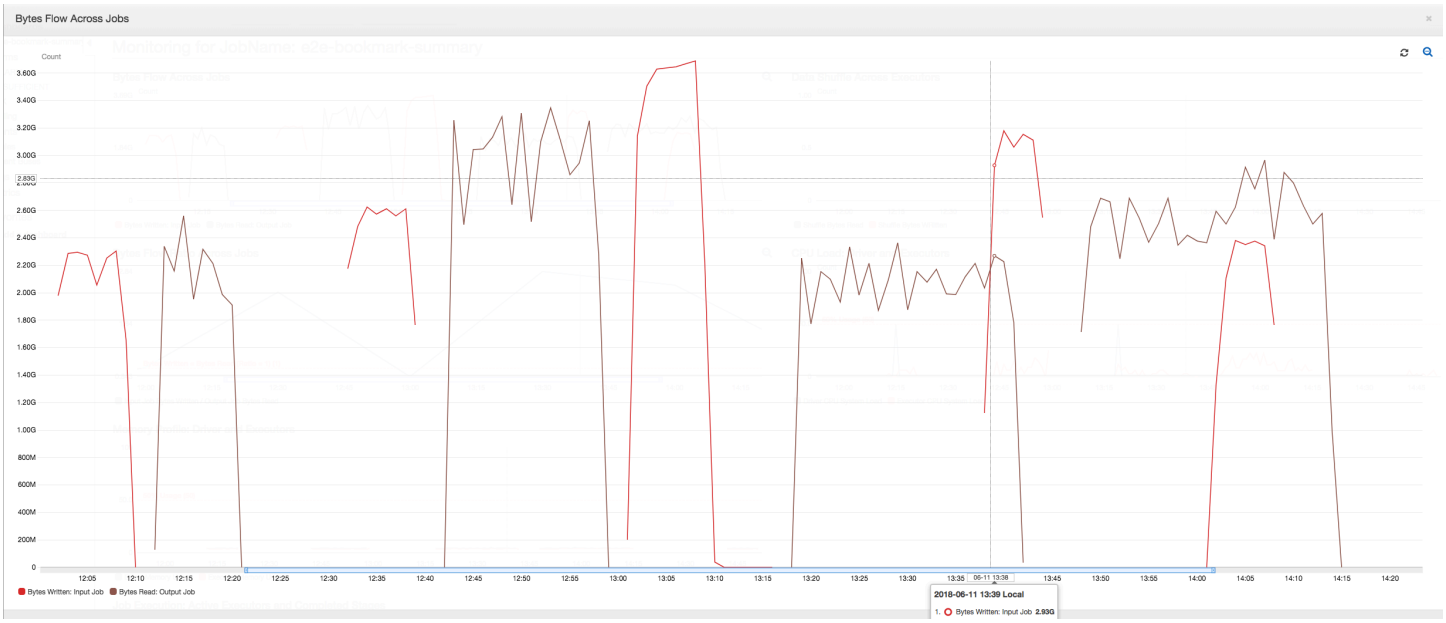
Ausgangsauftrag: Dieser Auftrag liest die Ausgabe des Eingangsauftrags vom Staging-Speicherort in Amazon S3, transformiert sie wieder und schreibt sie an ein Ziel:

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
```

```
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format = "json")
```

Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole

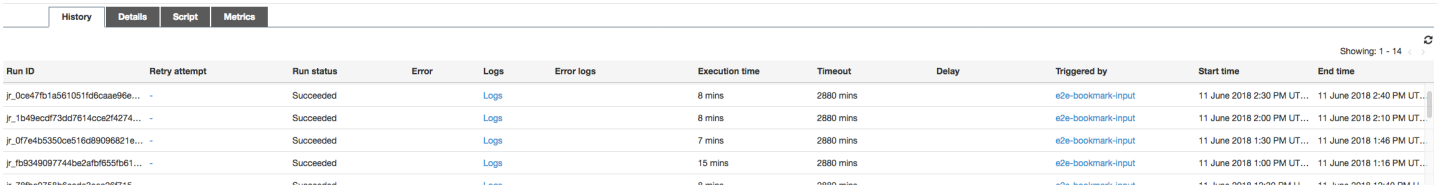
Das folgende Dashboard überlagert die geschriebenen Bytes der Amazon-S3-Metrik vom Eingangsauftrag mit den gelesenen Bytes der Amazon-S3-Metrik auf derselben Zeitachse für den Ausgangsauftrag. Die Zeitachse zeigt verschiedenen Auftragsausführungen der Eingangs- und Ausgangsaufträge. Der Eingangsauftrag (rot markiert) startet alle 30 Minuten. Der Ausgangsauftrag (braun dargestellt) beginnt bei Abschluss des Eingangsauftrags mit einer maximalen Nebenläufigkeit von 1.



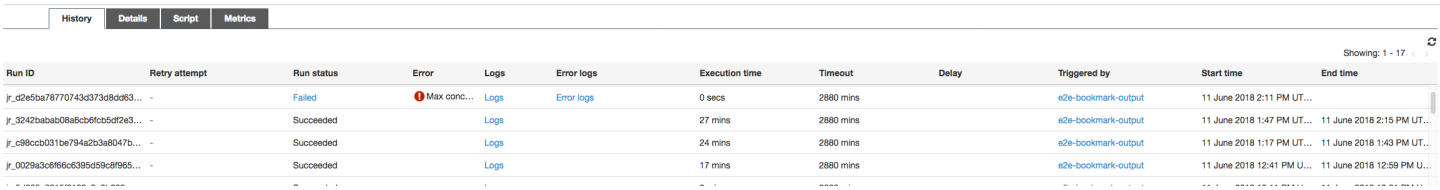
In diesem Beispiel sind die Auftragslesezeichen nicht aktiviert. Es werden keine Transformationskontexte verwendet, um Auftragslesezeichen im Skriptcode zu aktivieren.

Job History (Auftragsverlauf): Die Eingangs- und Ausgangsaufträge haben mehrere Ausführungen, wie auf der Registerkarte History (Verlauf) gezeigt, beginnend um 12:00 PM.

Der Eingabejob auf dem AWS Glue Die Konsole sieht so aus:

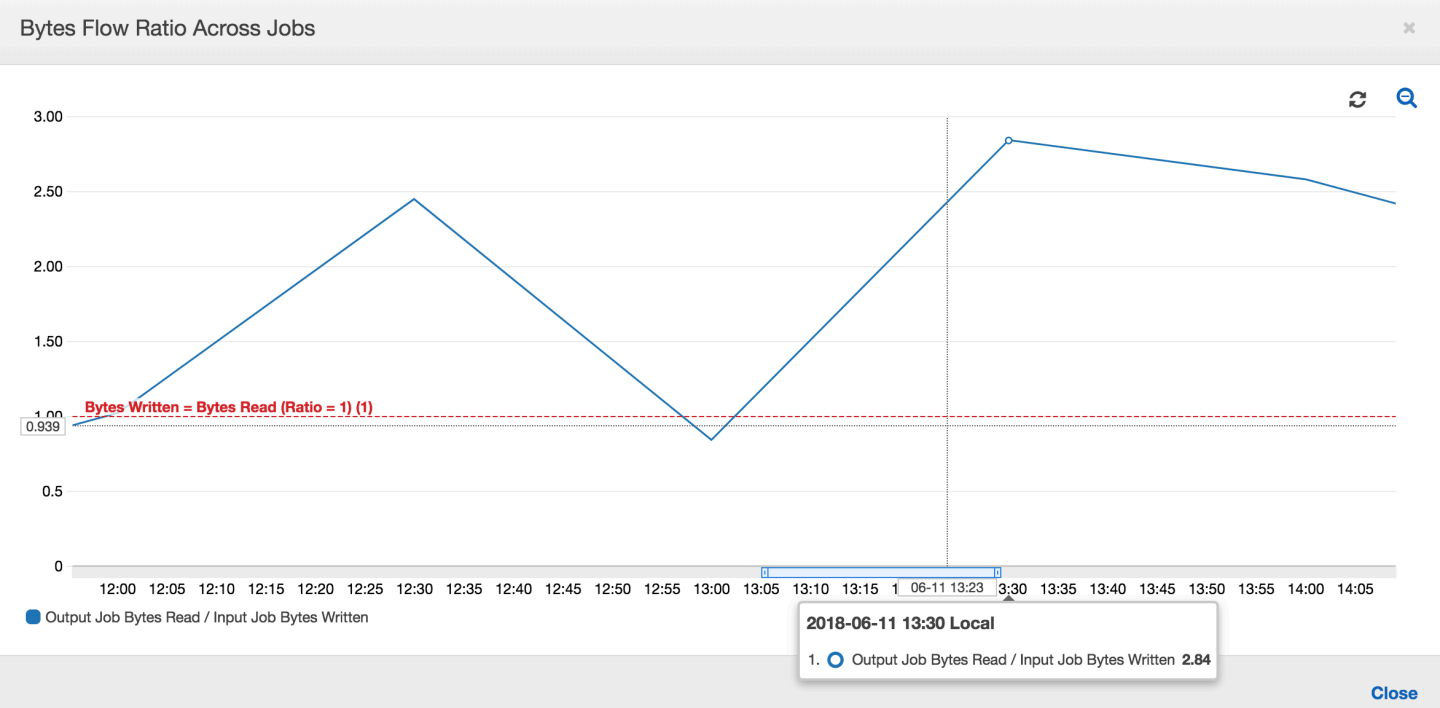


Das folgende Bild zeigt den Ausgangsauftrag:



First job runs (Erste Auftragsausführungen): Wie im folgenden Graphen für gelesene und geschriebene Datenbytes dargestellt, zeigen die ersten Auftragsausführungen der Eingangs- und Ausgangsaufträge zwischen 12:00 und 12:30 Uhr ungefähr die gleiche Fläche unter den Kurven. Diese Flächen stellen die vom Eingangsauftrag geschriebenen Amazon-S3-Bytes und die vom Ausgangsauftrag gelesenen Amazon-S3-Bytes dar. Diese Daten werden auch durch das Verhältnis von geschriebenen Amazon-S3-Bytes (summiert über 30 Minuten – die Auftragsauslösefrequenz für den Eingangsauftrag) bestätigt. Der Datenpunkt für das Verhältnis für die Eingangsauftragsausführung, die um 12:00 Uhr PM gestartet wurde, ist ebenfalls 1.

Der folgende Graph zeigt das Datenflussverhältnis für alle Auftragsausführungen:



Second job runs (Zweite Auftragsausführungen): In der zweiten Auftragsausführung gibt es eine klare Differenz zwischen der Anzahl der vom Ausgangsauftrag gelesenen Bytes im Vergleich zu der Anzahl der vom Eingangsauftrag geschriebenen Bytes. (Vergleichen Sie die Fläche unter der Kurve der beiden Auftragsausführungen für den Ausgangsauftrag oder vergleichen Sie die Flächen der zweiten Ausführung der Eingangs- und Ausgangsaufträge.) Das Verhältnis der gelesenen und geschriebenen Bytes zeigt, dass der Ausgangsauftrag in der zweiten Spanne von 30 Minuten von 12:30 bis

13:00 Uhr etwa die 2,5-fache Menge der vom Eingangsauftrag geschriebenen Daten gelesen hat. Der Grund hierfür ist, dass der Ausgangsauftrag die Ausgabe der ersten Auftragsausführung des Eingangsauftrags erneut verarbeitet hat, weil keine Auftragslesezeichen aktiviert waren. Ein Verhältnis von über 1 zeigt, dass es einen zusätzlichen Rückstand an Daten gibt, die von dem Ausgangsauftrag verarbeitet wurde.

Third job runs (Dritte Auftragsausführungen): Der Eingangsauftrag ist relativ konsistent in Bezug auf die Anzahl der geschriebenen Bytes (siehe Fläche unter der roten Kurven). Die dritte Auftragsausführung des Eingangsauftrags ist jedoch länger gelaufen als erwartet (siehe langer Auslauf der roten Kurve). Dies hat zur Folge, dass die dritte Auftragsausführung des Ausgangsauftrags zu spät gestartet wurde. Die dritte Auftragsausführung verarbeitet nur ein Bruchteil der akkumulierten Daten am Staging-Speicherort in den verbleibenden 30 Minuten zwischen 13:00 und 13:30 Uhr. Das Verhältnis des Bytestroms zeigt, dass sie nur 0,83 der von der dritten Auftragsausführung des Eingabeauftrags geschriebenen Daten verarbeitet hat (siehe Verhältnis um 13.00 Uhr).

Overlap of Input and Output jobs (Überlappende Eingangs- und Ausgangsaufträge): Die vierte Auftragsausführung des Eingangsauftrags startete um 13:30 gemäß den Zeitplan, bevor die dritte Auftragsausführung des Ausgangsauftrags abgeschlossen wurde. Es gibt eine partielle Überschneidung zwischen diesen beiden Auftragsausführungen. Die dritte Auftragsausführung des Ausgangsauftrags erfasst nur die Dateien, die am Staging-Speicherort von Amazon S3 aufgelistet sind, als sie etwa um 13:17 Uhr begonnen hat. Dies umfasst alle Datenausgaben aus den ersten Auftragsausführungen des Eingangsauftrags. Das tatsächliche Verhältnis um 13:30 ist etwa 2,75. Die dritte Auftragsausführung des Ausgangsauftrags hat etwa das 2,75-fache der von der vierten Auftragsausführung von 13:30 bis 14:00 Uhr geschriebenen Daten verarbeitet.

Wie diese Bilder zeigen, verarbeitet der Ausgangsauftrag Daten vom Staging-Standort aus allen vorherigen Auftragsausführungen des Eingangsauftrags erneut. Dies hat zur Folge, dass die vierte Auftragsausführung für den Ausgangsauftrag die längste ist und sich mit der gesamten fünften Auftragsausführung des Eingangsauftrags überlappt.

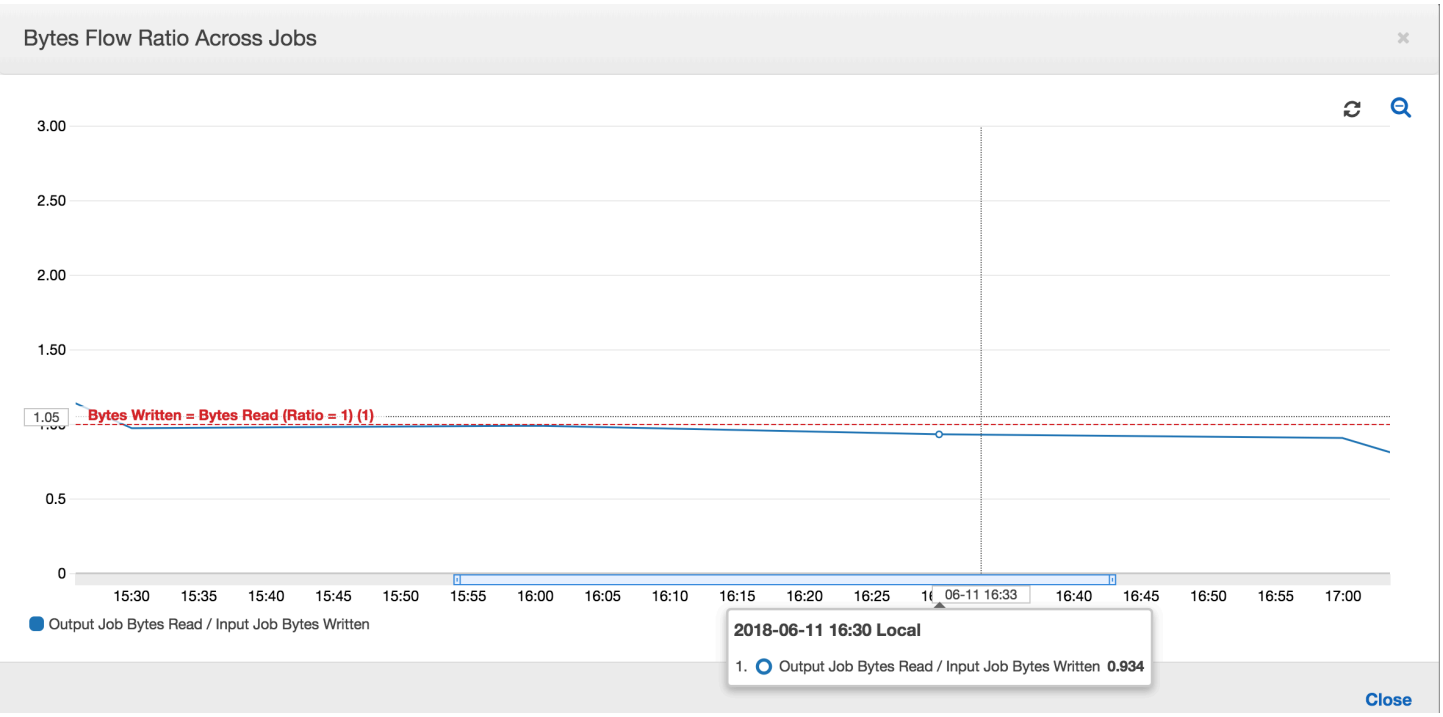
Korrektur der Dateiverarbeitung

Sie sollten sicherstellen, dass der Ausgangsauftrag nur die Dateien verarbeitet, die von den vorherigen Auftragsausführungen des Ausgangsauftrags nicht verarbeitet wurden. Dazu aktivieren Sie die Auftragslesezeichen und legen Sie den Transformationskontext im Ausgangsauftrag wie folgt fest:

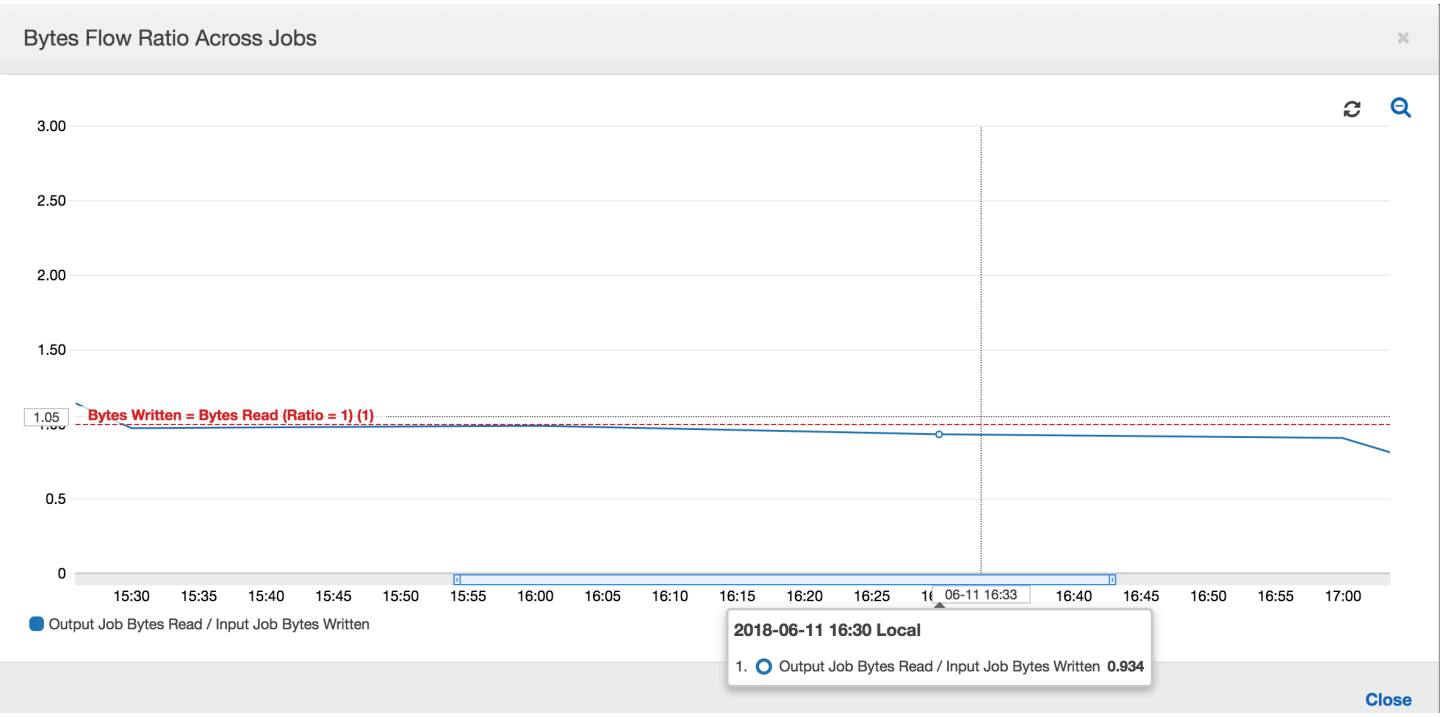
```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json", transformation_ctx =
 "bookmark_ctx")
```

Bei aktivierten Auftragslesezeichen verarbeitet der Ausgangsauftrag die Daten am Staging-Speicherort aus allen vorherigen Auftragsausführungen des Eingangsauftrags nicht erneut. In der folgenden Abbildung mit den gelesenen und geschriebenen Daten ist die Fläche unter der braunen Kurve relativ konsistent und ähnlich den roten Kurven.

Auch die Verhältnisse des Byteflusses bleiben etwa bei 1, da keine zusätzlichen Daten verarbeitet werden.



Eine Auftragsausführung für den Ausgangsauftrag startet und erfasst die Dateien am Staging-Speicherort, bevor die nächste Eingangsauftragsausführung gestartet wird, womit weitere Daten am Staging-Standort abgelegt werden. Solange dies fortgesetzt wird, verarbeitet sie nur die Dateien, die von der vorherigen Eingangsauftragsausführung erfasst wurden, und das Verhältnis bleibt etwa 1.



Angenommen, der Eingangsauftrag dauert länger als erwartet, und der Ausgangsauftrag erfasst aus diesem Grund Dateien am Staging-Standort von zwei Eingangsauftragsausführungen. Das Verhältnis wird dann höher als 1 für diese Ausgangsauftragausführung. Die folgenden Auftragsausführungen des Ausgangsauftrags verarbeiten jedoch keine Dateien, die bereits von den vorherigen Auftragsausführungen des Ausgangsauftrags verarbeitet wurden.

Überwachung für die DPU-Kapazitätsplanung

Sie können Job-Metriken verwenden AWS Glue , um die Anzahl der Datenverarbeitungseinheiten (DPUs) zu schätzen, die zur Skalierung eines AWS Glue Job.

> **Note**
>
> Diese Seite gilt nur für AWS Glue Versionen 0.9 und 1.0. Spätere Versionen von AWS Glue enthalten kostensparende Funktionen, die zusätzliche Überlegungen bei der Kapazitätsplanung mit sich bringen.

Themen

- Profilierter Code
- Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole
- Bestimmen der optimalen DPU-Kapazität

Profilierter Code

Das folgende Skript liest eine Amazon Simple Storage Service (Amazon S3)-Partition mit 428 gezippten JSON-Dateien. Das Skript wendet ein Mapping an, um die Feldnamen zu ändern, und konvertiert und schreibt sie in Amazon S3 im Apache-Parquet-Format. Sie stellen 10 DPUs standardmäßig bereit und führen diesen Job aus.

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [input_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [(map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format =
 "parquet")
```

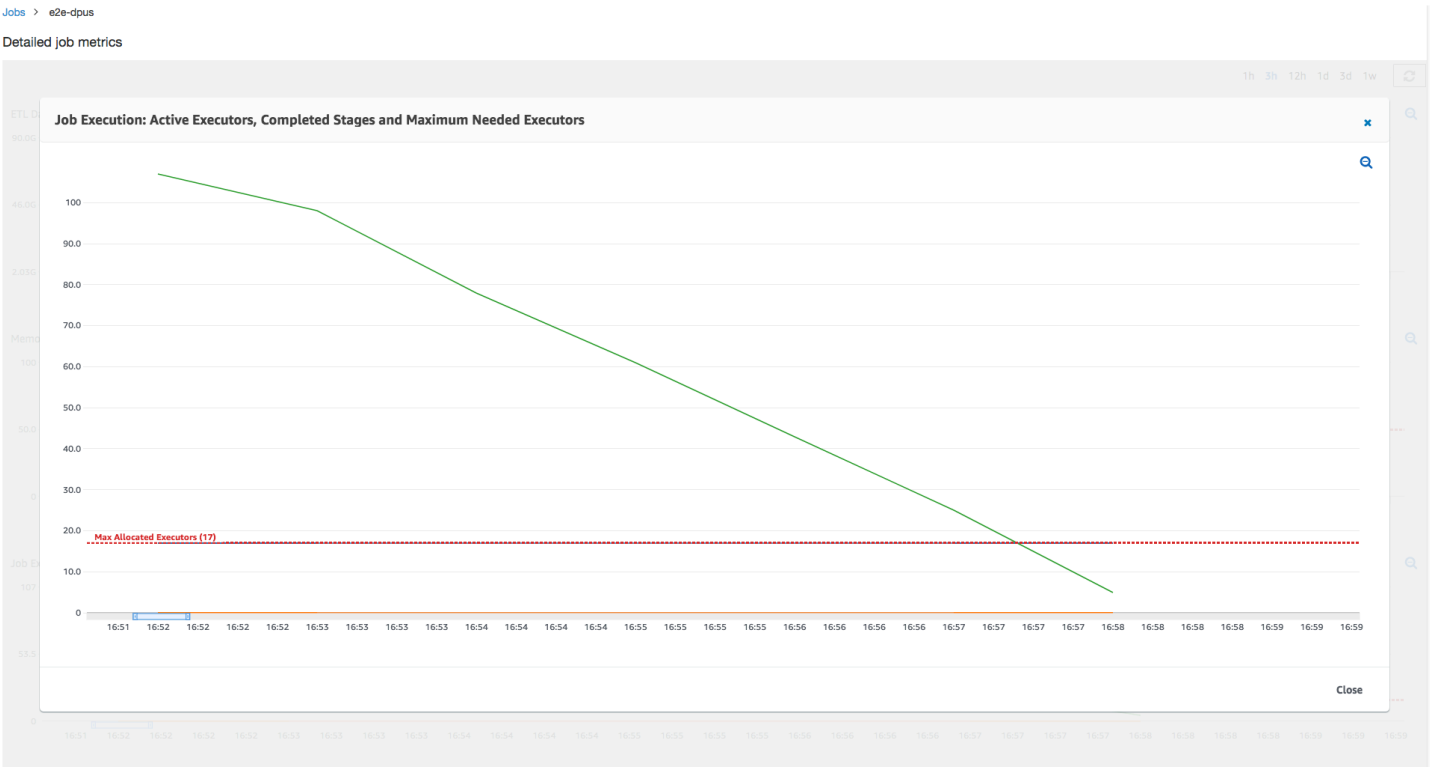
Visualisieren Sie die profilierten Metriken auf der AWS Glue Konsole

Joblauf 1: In diesem Job-Lauf zeigen wir, wie Sie herausfinden können, ob DPUs im Cluster zu wenig Provisioning vorhanden ist. Die Funktionalität zur Auftragsausführung in AWS Glue zeigt die Gesamtzahl der aktiv ausgeführten Executoren, die Anzahl der abgeschlossenen Phasen und die Anzahl der maximal benötigten Executoren an.

Die Anzahl der maximal benötigten Executors wird berechnet, indem die Gesamtzahl der laufenden Aufgaben und anstehenden Aufgaben addiert und durch die Aufgaben pro Executor dividiert wird. Dieses Ergebnis ist ein Maß für die Gesamtanzahl der Executors, die erforderlich sind, um die aktuelle Last zu erfüllen.

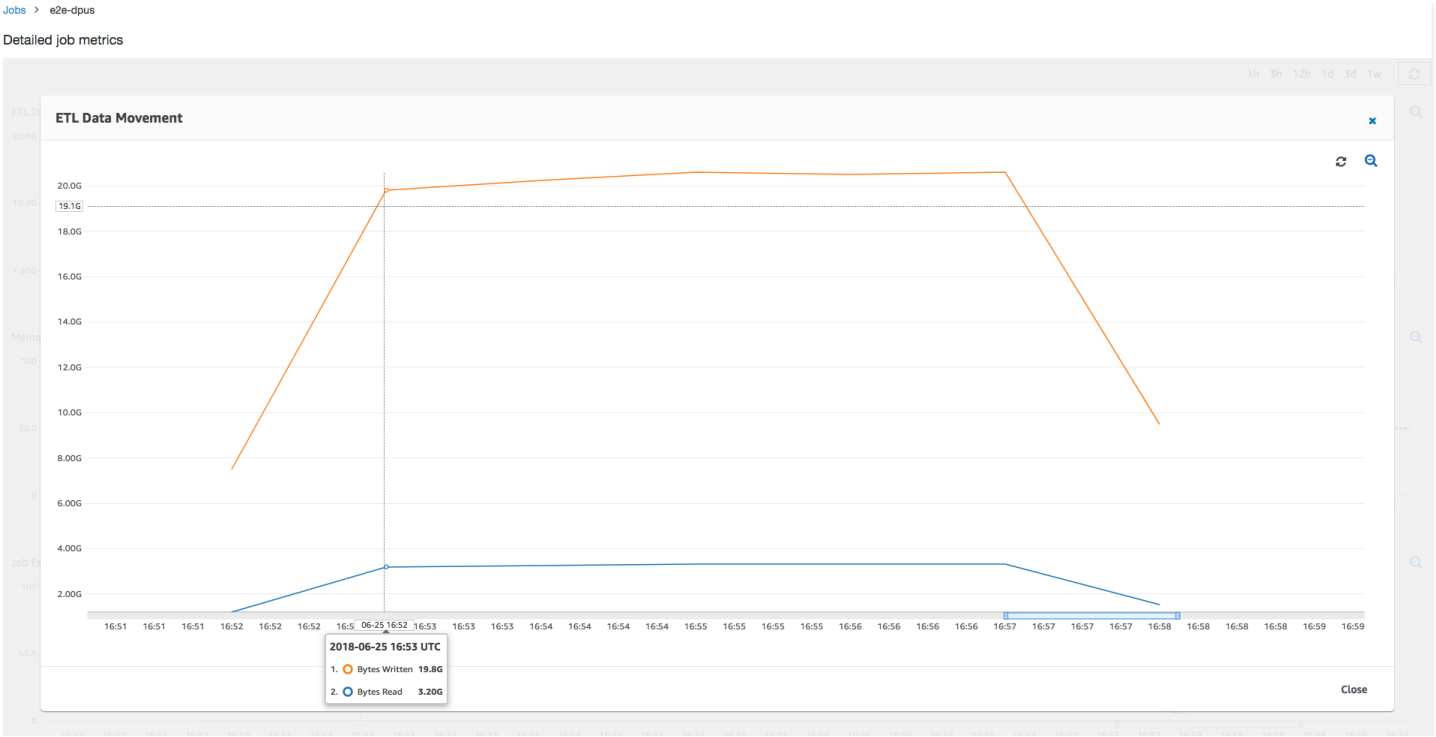
Im Gegensatz dazu misst die Anzahl der aktiven Executors, wie viele aktive Apache-Spark-Aufgaben ausgeführt werden. Im weiteren Verlauf des Auftrags können sich die maximal benötigten Executors ändern und gehen in der Regel gegen Ende des Auftrags zurück, da die Warteschlange der anstehenden Aufgaben abnimmt.

Die horizontale rote Linie in der folgenden Grafik zeigt die Anzahl der maximal zugewiesenen Executoren, die von der Anzahl der Executoren abhängt, die Sie dem Job DPUs zuweisen. In diesem Fall weisen Sie dem ausgeführten Job 10 DPUs zu. Eine DPU ist für die Verwaltung reserviert. Neun DPUs führen jeweils zwei Executors aus und ein Executor ist für den Spark-Treiber reserviert. Der Spark-Treiber wird in der primären Anwendung ausgeführt. Die Anzahl der maximalen zugewiesenen Executors ist also 2* 9 – 1 = 17 Executors.



Wie der Graph zeigt, beginnt die Anzahl der maximal benötigten Executors bei 107 zu Beginn des Auftrags, während die Anzahl der aktiven Executors bei 17 bleibt. Dies entspricht der Anzahl der maximal zugewiesenen Executoren mit 10. DPUs Das Verhältnis zwischen den maximal benötigten Executors und den maximal zugewiesenen Executors (zu beiden wird für den Spark-Treiber 1 addiert) ergibt den Faktor für die zu niedrig ausgelegte Bereitstellung: 108/18 = 6x. Sie können 6 (unter dem Provisioning Ratio) *9 (aktuelle DPU-Kapazität — 1) + 1 DPUs = 55 bereitstellen, um den Job so DPUs zu skalieren, dass er mit maximaler Parallelität ausgeführt und schneller abgeschlossen wird.

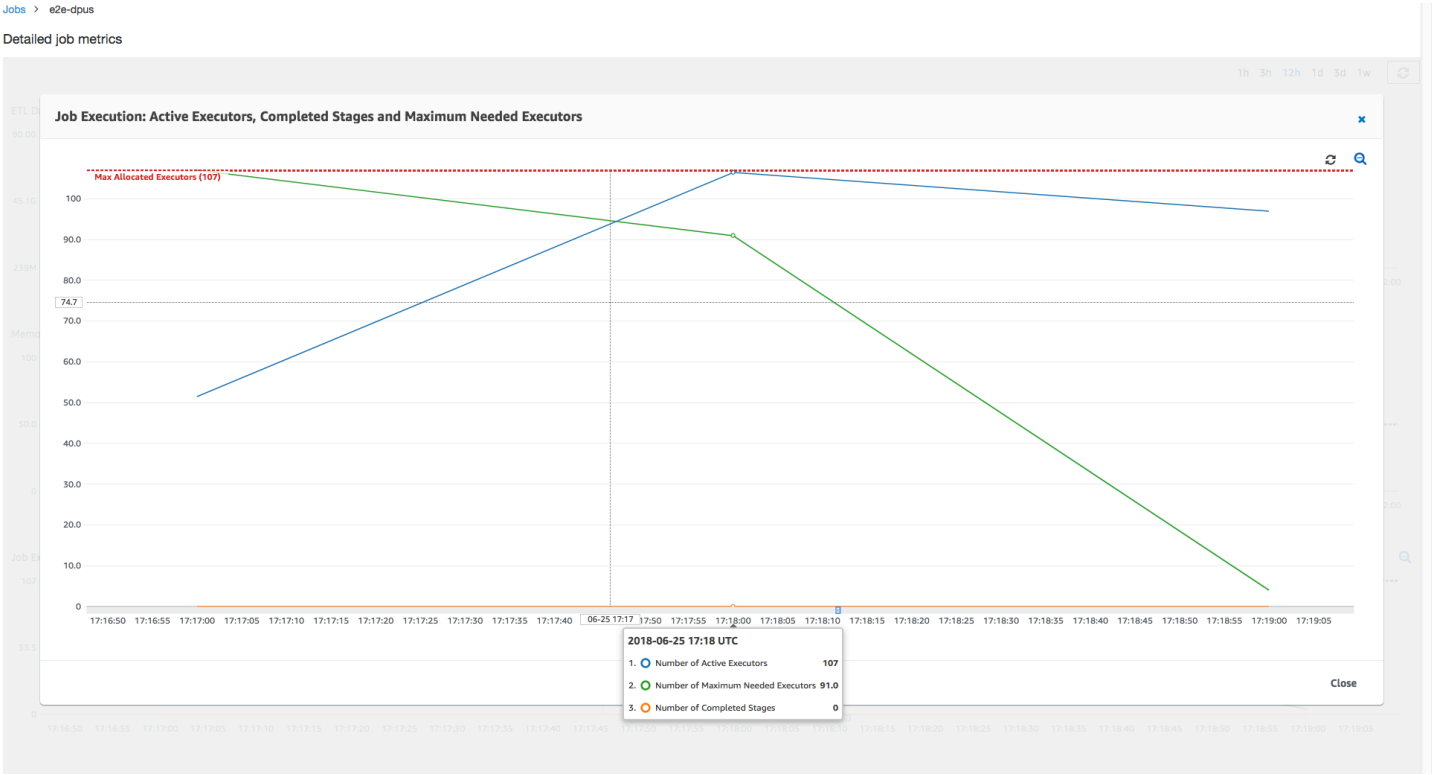
Die AWS Glue -Konsole zeigt die detaillierten Auftragsmetriken als statische Linie an, die die ursprüngliche Anzahl der maximal zugeteilten Executors darstellt. Die Konsole berechnet die maximal zugeteilten Executors aus der Auftragsdefinition für die Metriken. Im Gegensatz dazu berechnet die Konsole für detaillierte Messwerte zur Auftragsausführung die maximal zugewiesenen Executoren aus der Konfiguration der Auftragsausführung, insbesondere die für die Auftragsausführung zugewiesenen. DPUs Um Metriken für eine einzelne Auftragsausführung anzuzeigen, wählen Sie die Auftragsausführung und anschließend View run metrics (Ausführungsmetriken anzeigen) aus.



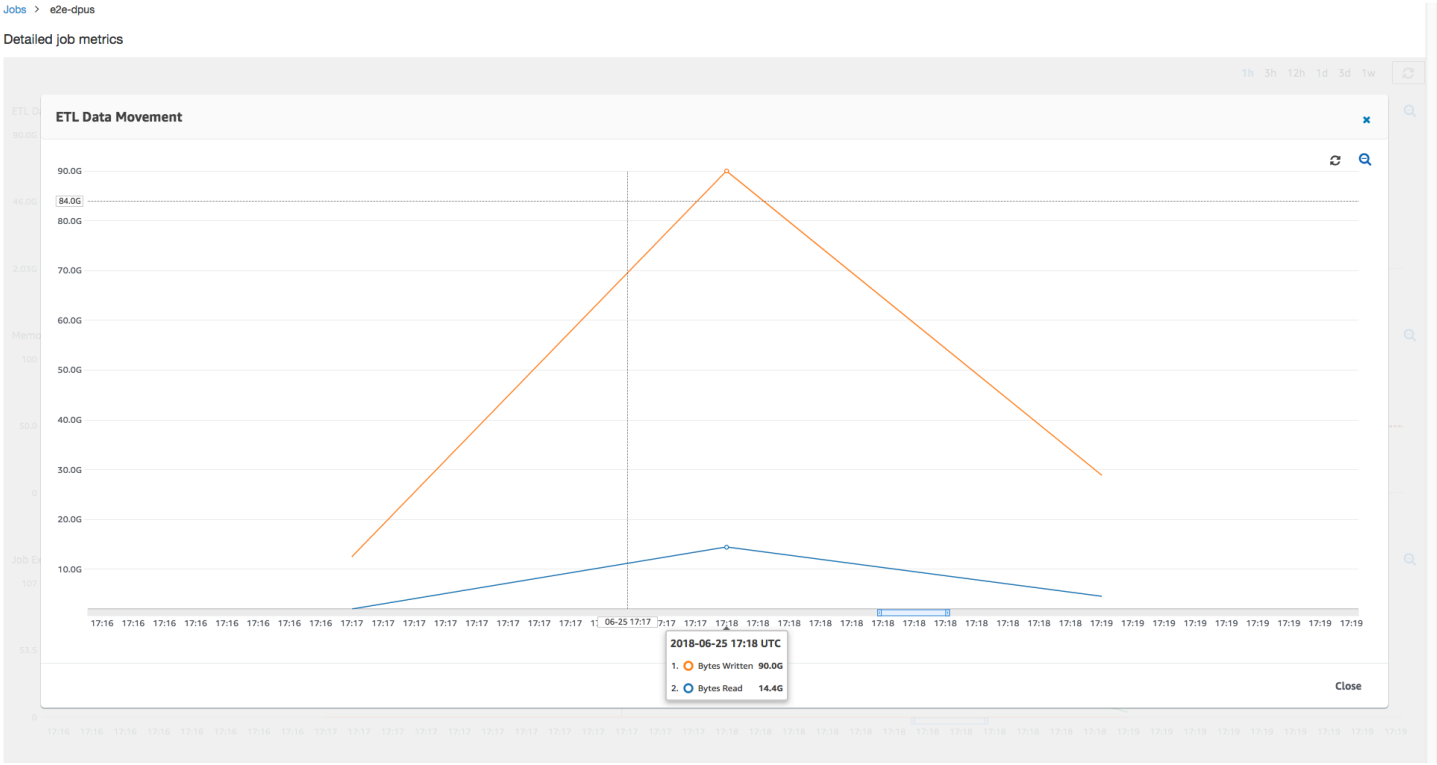
Bei der Betrachtung der gelesenen und geschriebenen Amazon-S3-Bytes erkennen Sie, dass der Auftrag alle sechs Minuten parallel Daten aus Amazon S3 streamt und schreibt. Alle Kerne auf den zugewiesenen DPUs Kernen lesen und schreiben in Amazon S3. Die maximale Anzahl an benötigten Executors von 107 entspricht auch der Anzahl der Dateien im Amazon-S3-Eingangspfad 428. Jeder Executor können vier Spark-Aufgaben zur Verarbeitung von vier Eingangsdateien (gezippte JSON-Dateien) starten.

Bestimmen der optimalen DPU-Kapazität

Basierend auf den Ergebnissen der vorherigen Auftragsausführung können Sie die Gesamtzahl der zugewiesenen Aufgaben DPUs auf 55 erhöhen und so die Leistung des Auftrags überprüfen. Der Auftrag wird in weniger als drei Minuten abgeschlossen – der Hälfte der Zeit, die zuvor erforderlich war. Die Auftragsskalierung erfolgt in diesem Fall nicht linear, da es sich um einen kurzen Auftrag handelt.. Jobs mit langlebigen Aufgaben oder einer großen Anzahl von Aufgaben (eine große Anzahl von maximal benötigten Executoren) profitieren von einer close-to-linear DPU-Scale-Out-Leistungssteigerung.



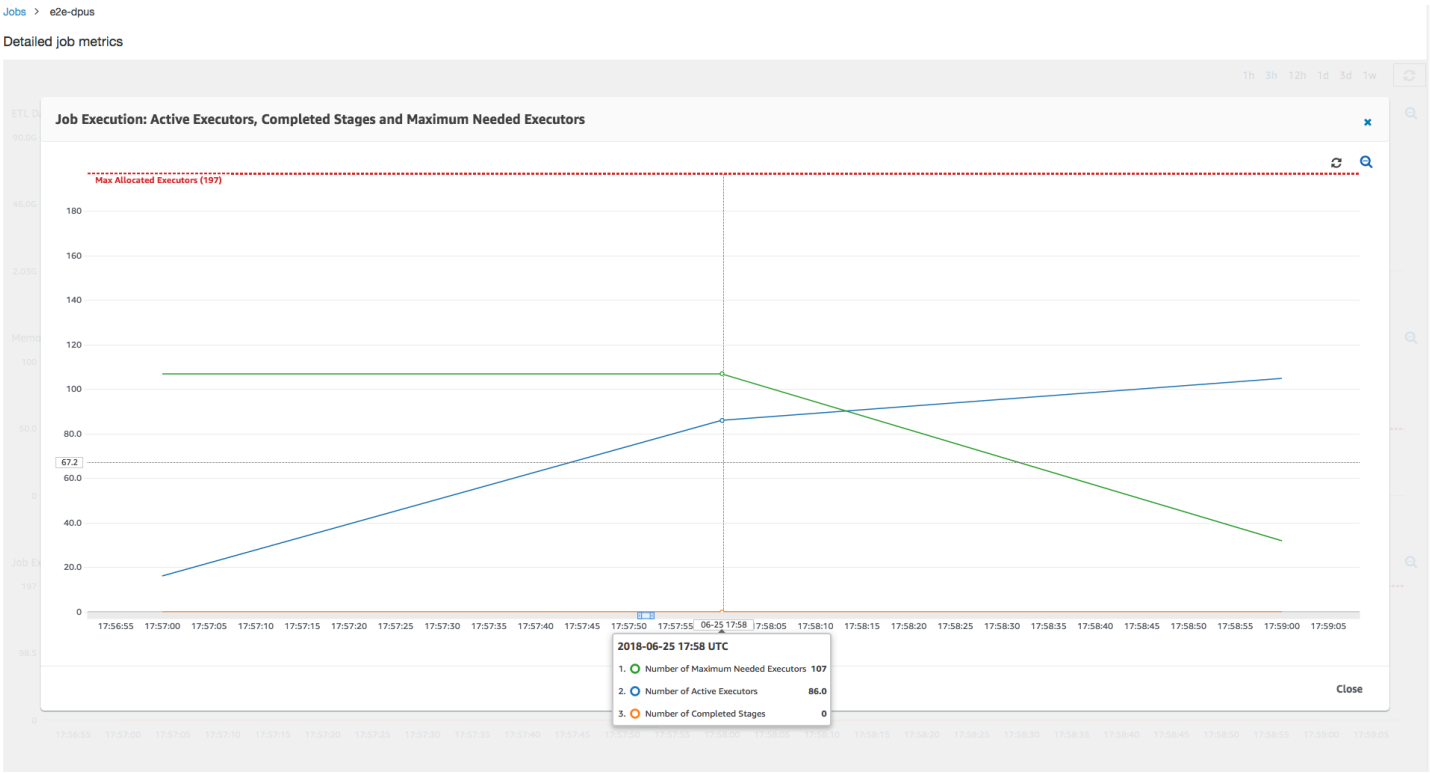
Wie die folgende Abbildung zeigt, erreicht die Gesamtzahl der aktiven Executors die maximale Anzahl zugewiesener Executors107. Ebenso liegt die maximale Anzahl benötigter Executors nie über den maximal zugewiesenen Executors. Die maximale Anzahl erforderlicher Executors wird anhand der Anzahl der aktiven und ausstehenden Aufgaben berechnet, sie könnte also kleiner sein als die Anzahl der aktiven Executors. Dies liegt daran, dass es Executors geben kann, die für einen kurzen Zeitraum teilweise oder vollständig im Leerlauf sind und noch nicht außer Betrieb genommen wurden.



Dieser Auftrag nutzt sechsmal mehr Executors für parallele Lese- und Schreibvorgänge in Amazon S3. Dies hat zur Folge, dass diese Auftragsausführung mehr Amazon-S3-Bandbreite für Lese- und Schreibvorgänge verwendet und schneller fertiggestellt wird.

Identifizieren Sie übermäßig bereitgestellte DPUs

Als Nächstes können Sie feststellen, ob eine weitere Skalierung des Jobs mit 100 DPUs (99 * 2 = 198 Executoren) hilfreich ist. Wie der folgende Graph zeigt, dauert die Fertigstellung des Auftrags immer noch drei Minuten. Ebenso kann der Job nicht über 107 Executoren (DPUsKonfiguration 55) hinaus skaliert werden, und die verbleibenden 91 Executoren sind überprovisioniert und werden überhaupt nicht verwendet. Dies zeigt, dass eine Erhöhung der Anzahl von Executoren DPUs möglicherweise nicht immer zu einer Verbesserung der Leistung führt, wie anhand der maximal benötigten Anzahl von Executoren deutlich wird.



Vergleich von Zeitunterschieden

Die drei in der folgenden Tabelle aufgeführten Auftragsausführungen fassen die Ausführungszeiten für 10 DPUs DPUs, 55 und 100 DPUs Jobs zusammen. Sie finden die DPU-Kapazität zur Verbesserung der Auftragsausführungszeit unter Verwendung der Schätzungen, die Sie bei der Überwachung der ersten Auftragsausführung eingerichtet haben.

Auftrags-ID	Anzahl der DPUs	Execution time (Ausführu ngszeit)
jr_c894524c8ef5048a4d9...	10	6 Minuten
jr_1a466cf2575e7ffe6856...	55	3 Minuten
jr_34fa1ed4c6aa9ff0a814...	100	3 Minuten

Generative KI-Fehlerbehebung für Apache Spark in AWS Glue

Die generative KI-Fehlerbehebung für die Apache Spark-Vorschau ist für Jobs verfügbar, die auf AWS Glue 4.0 und AWS Glue 5.0 ausgeführt werden, und in den folgenden AWS Regionen:

USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), USA West (Nordkalifornien), Südamerika (São Paulo), Kanada (Zentral), Europa (Irland), Europa (London), Europa (Paris), Europa (Stockholm), Europa (Mailand), Europa (Frankfurt), Naher Osten (Bahrain), Naher Osten (VAE), Afrika (Kapstadt), Asien-Pazifik (Tokio), Asien-Pazifik (Hongkong), Asien-Pazifik (Mumbai), Asien-Pazifik (Singapur), Asien-Pazifik (Jakarta), Asien-Pazifik (Seoul), Asien-Pazifik (Osaka) und Asien-Pazifik (Sydney).

Die Vorschaufunktionen können sich ändern.

Generative KI-Fehlerbehebung für Apache Spark-Jobs in AWS Glue ist eine neue Funktion, mit der Dateningenieure und Wissenschaftler Probleme in ihren Spark-Anwendungen mühelos diagnostizieren und beheben können. Mithilfe von maschinellem Lernen und generativen KI-Technologien analysiert diese Funktion Probleme in Spark-Jobs und bietet eine detaillierte Ursachenanalyse sowie umsetzbare Empfehlungen zur Lösung dieser Probleme.

Wie funktioniert Generative KI Troubleshooting für Apache Spark?

Für Ihre fehlgeschlagenen Spark-Jobs analysiert Generative AI Troubleshooting die Job-Metadaten und die genauen Metriken und Protokolle, die mit der Fehlersignatur Ihres Jobs verknüpft sind, um eine Ursachenanalyse zu erstellen, und empfiehlt spezifische Lösungen und bewährte Verfahren zur Behebung von Jobfehlern.

Einrichtung von Generative KI Troubleshooting für Apache Spark für Ihre Jobs

Konfigurieren von IAM-Berechtigungen

Um Benutzern, die von Spark Troubleshooting für Ihre Jobs in AWS Glue APIs verwendet werden, Berechtigungen zu gewähren, sind entsprechende IAM-Berechtigungen erforderlich. Sie können Berechtigungen erhalten, indem Sie Ihrer IAM-Identität (z. B. einem Benutzer, einer Rolle oder einer Gruppe) die folgende benutzerdefinierte AWS Richtlinie zuordnen.

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
"Effect": "Allow",
```

```
            "Action": [
                "glue:StartCompletion",
                "glue:GetCompletion"
            ],
            "Resource": [
                "arn:aws:glue:*:*:completion/*",
                "arn:aws:glue:*:*:job/*"
            ]
        }
    ]
}
```

> **Note**
>
> In der Vorschauversion ist Spark Troubleshooting nicht über das AWS SDK APIs verfügbar, das Sie programmgesteuert verwenden können. Die folgenden beiden APIs werden in der IAM-Richtlinie verwendet, um diese Erfahrung über die AWS Glue Studio-Konsole zu aktivieren: `StartCompletion` und`GetCompletion`.

Zuweisen von Berechtigungen

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

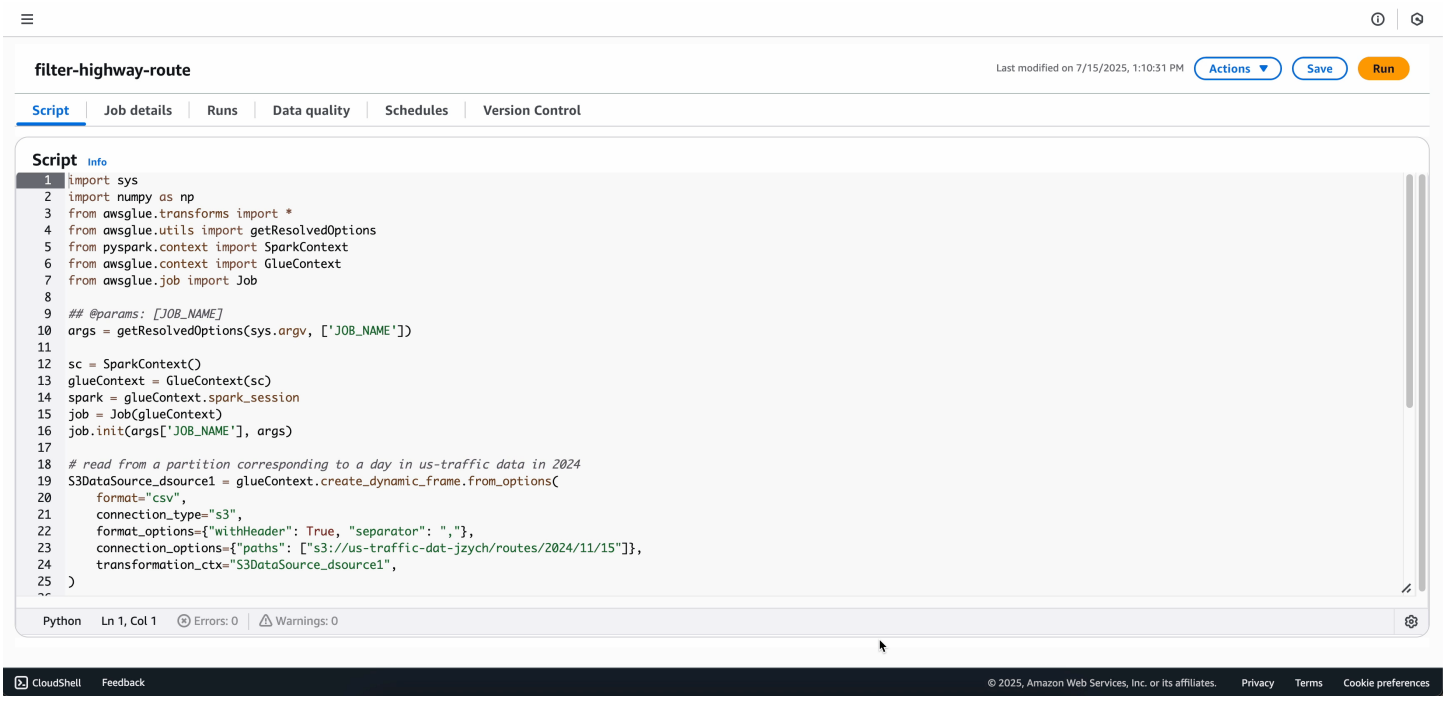
- Für Benutzer und Gruppen in IAM Identity Center: Erstellen Sie einen Berechtigungssatz. Folgen Sie den Anweisungen unter Einen Berechtigungssatz erstellen im IAM Identity Center-Benutzerhandbuch.
- Für Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden: Erstellen Sie eine Rolle für den Identitätsverbund. Folgen Sie den Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Federation) im IAM-Benutzerhandbuch.
- Für IAM-Benutzer: Erstellen Sie eine Rolle, die Ihr Benutzer übernehmen kann. Folgen Sie den Anweisungen unter Eine Rolle für einen IAM-Benutzer erstellen im IAM-Benutzerhandbuch.

Die Analyse zur Problembehandlung wird anhand einer fehlgeschlagenen Jobausführung ausgeführt

Sie können über mehrere Pfade in der AWS Glue-Konsole auf die Fehlerbehebungsfunktion zugreifen. So können Sie loslegen:

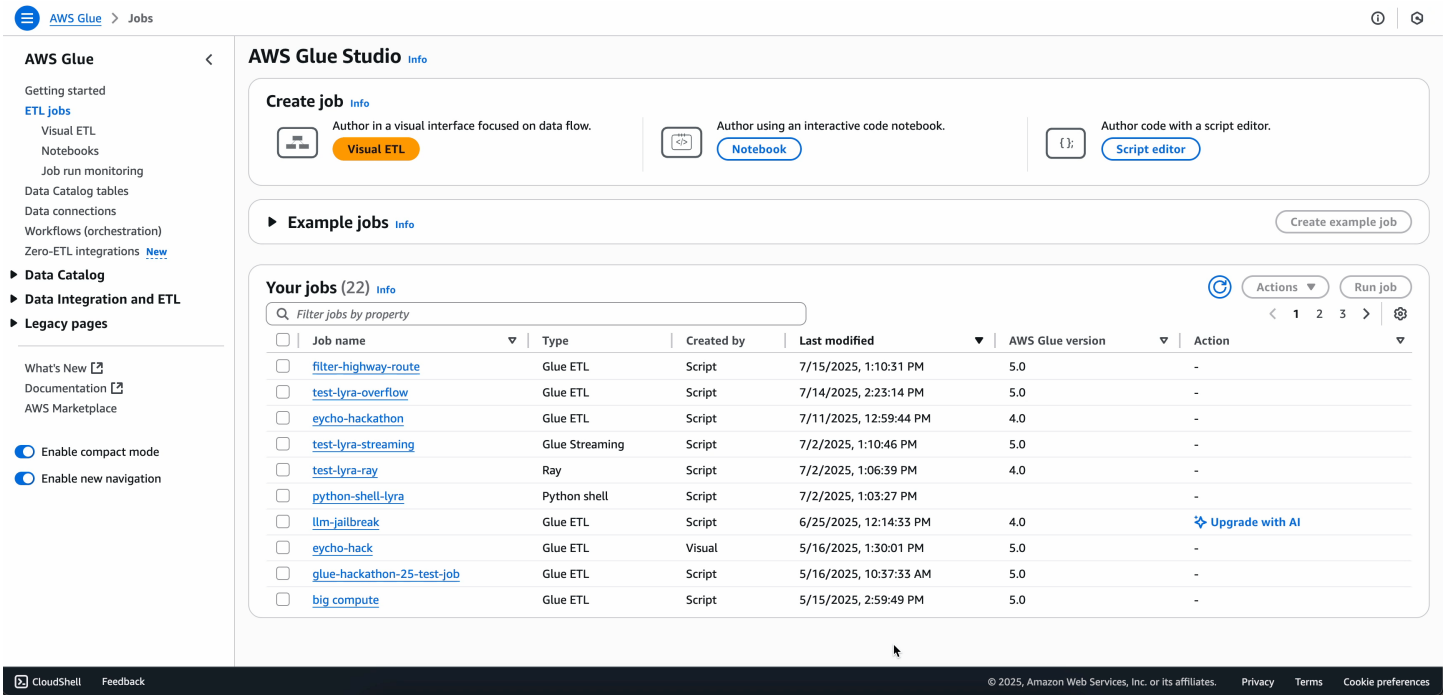
Option 1: Auf der Seite mit der Jobliste

1. Öffnen Sie die AWS Glue-Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich ETL Jobs aus.
3. Suchen Sie Ihren fehlgeschlagenen Job in der Jobliste.
4. Wählen Sie im Abschnitt mit den Auftragsdetails die Registerkarte Läufe aus.
5. Klicken Sie auf die fehlgeschlagene Auftragsausführung, die Sie analysieren möchten.
6. Wählen Sie Troubleshooting with AI, um die Analyse zu starten.
7. Wenn die Analyse zur Fehlerbehebung abgeschlossen ist, können Sie die Ursachenanalyse und die Empfehlungen auf der Registerkarte Analyse zur Fehlerbehebung am unteren Bildschirmrand einsehen.



Option 2: Verwenden der Seite „Job Run Monitoring“

1. Navigieren Sie zur Seite zur Überwachung der Jobausführung.
2. Suchen Sie nach Ihrer fehlgeschlagenen Auftragsausführung.
3. Wählen Sie das Drop-down-Menü Aktionen aus.
4. Wählen Sie Problembehandlung mit KI.



Option 3: Von der Seite mit den Job-Ausführungsdetails

1. Navigieren Sie zur Detailseite Ihres fehlgeschlagenen Auftragslaufs, indem Sie entweder auf der Registerkarte Ausführungen auf Details zu einem fehlgeschlagenen Lauf anzeigen klicken oder den ausgeführten Job auf der Seite zur Überwachung des Auftragslaufs auswählen.
2. Suchen Sie auf der Detailseite des Auftragslaufs die Registerkarte Analyse zur Fehlerbehebung.

Unterstützte Kategorien zur Fehlerbehebung (Vorschau)

Dieser Service konzentriert sich auf drei Hauptkategorien von Problemen, auf die Dateningenieure und Entwickler in ihren Spark-Anwendungen häufig stoßen:

- Fehler bei der Einrichtung und beim Zugriff auf Ressourcen: Beim Ausführen von Spark-Anwendungen in AWS Glue gehören Fehler bei der Einrichtung und beim Zugriff auf Ressourcen zu den häufigsten, aber schwierig zu diagnostizierenden Problemen. Diese Fehler treten häufig auf, wenn Ihre Spark-Anwendung versucht, mit AWS Ressourcen zu interagieren, aber auf Berechtigungsprobleme, fehlende Ressourcen oder Konfigurationsprobleme stößt.
- Speicherprobleme mit Spark-Treibern und Executoren: Speicherbezogene Fehler in Apache Spark-Jobs können komplex zu diagnostizieren und zu beheben sein. Diese Fehler treten häufig auf, wenn Ihre Datenverarbeitungsanforderungen die verfügbaren Speicherressourcen überschreiten, entweder auf dem Treiberknoten oder auf den Executor-Knoten.

- Probleme mit der Spark-Festplattenkapazität: Speicherbedingte Fehler in AWS Glue Spark-Jobs treten häufig bei Shuffle-Vorgängen, beim Verschütten von Daten oder bei umfangreichen Datentransformationen auf. Diese Fehler können besonders knifflig sein, da sie möglicherweise erst auftreten, wenn Ihr Job eine Weile ausgeführt wurde, wodurch möglicherweise wertvolle Rechenzeit und Ressourcen verschwendet werden.

> **Note**
>
> Bevor Sie vorgeschlagene Änderungen in Ihrer Produktionsumgebung implementieren, sollten Sie die vorgeschlagenen Änderungen sorgfältig prüfen. Der Service bietet Empfehlungen, die auf Mustern und bewährten Methoden basieren. Ihr spezieller Anwendungsfall erfordert jedoch möglicherweise zusätzliche Überlegungen.

AWS Glue Typen von Arbeitern

Übersicht

AWS Glue bietet mehrere Workertypen, um unterschiedlichen Workload-Anforderungen gerecht zu werden, von kleinen Streaming-Jobs bis hin zu umfangreichen, speicherintensiven Datenverarbeitungsaufgaben. Dieser Abschnitt enthält umfassende Informationen zu allen verfügbaren Workertypen, ihren Spezifikationen und Verwendungsempfehlungen.

Kategorien der Arbeitnehmertypen

AWS Glue bietet zwei Hauptkategorien von Arbeitnehmertypen:

- G-Worker-Typen: Allzweck-Computing-Worker, die für Standard-ETL-Workloads optimiert sind
- R-Worker-Typen: Speicheroptimierte Worker, die für speicherintensive Spark-Anwendungen konzipiert wurden

Datenverarbeitungseinheiten () DPUs

Die den AWS Glue Arbeitnehmern zur Verfügung stehenden Ressourcen werden in gemessen DPUs. Eine DPU ist ein relatives Maß für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt.

Speicheroptimiert DPUs (M-DPUs): Worker vom Typ R verwenden M-DPUs, wodurch im Vergleich zum Standard die doppelte Speicherzuweisung für eine bestimmte Größe bereitgestellt wird. DPUs Das bedeutet, dass eine Standard-DPU 16 GB Speicher bereitstellt, während eine M-DPU in Workern vom Typ R 32 GB Speicher bereitstellt, der für speicherintensive Spark-Anwendungen optimiert ist.

Verfügbare Worker-Typen

G.1X - Standardarbeiter

- DPU: 1 DPU (4 VCPUs, 16 GB Speicher)
- Speicher: 94 GB Festplatte (ca. 44 GB frei)
- Anwendungsfall: Datentransformationen, Verknüpfungen und Abfragen — skalierbar und kostengünstig für die meisten Aufgaben

G.2X — Standardarbeiter

- DPU: 2 DPU (8 VCPUs, 32 GB Speicher)
- Speicher: 138 GB Festplatte (ca. 78 GB frei)
- Anwendungsfall: Datentransformationen, Verknüpfungen und Abfragen — skalierbar und kostengünstig für die meisten Aufgaben

G.4X — Großer Arbeiter

- DPU: 4 DPU (16 VCPUs, 64 GB Speicher)
- Speicher: 256 GB Festplatte (ca. 230 GB frei)
- Anwendungsfall: Anspruchsvolle Transformationen, Aggregationen, Verknüpfungen und Abfragen

G.8X — Sehr großer Arbeiter

- DPU: 8 DPU (32 VCPUs, 128 GB Speicher)
- Speicher: 512 GB Festplatte (ca. 485 GB frei)
- Anwendungsfall: Die anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen

G.12X — Sehr großer Arbeiter*

- DPU: 12 DPU (48 V, 192 GB Speicher) CPUs
- Speicher: 768 GB Festplatte (ca. 741 GB frei)
- Anwendungsfall: Sehr große und ressourcenintensive Workloads, die eine erhebliche Rechenkapazität erfordern

G.16X — Maximaler Mitarbeiter*

- DPU: 16 DPU (64 V, 256 GB Speicher) CPUs
- Speicher: 1024 GB Festplatte (ca. 996 GB frei)
- Anwendungsfall: Größte und ressourcenintensivste Workloads, die maximale Rechenkapazität erfordern

R.1X — Speicheroptimiert klein*

- DPU: 1 M-DPU (4 V, 32 GB Speicher) CPUs
- Anwendungsfall: Speicherintensive Workloads mit häufigen Fehlern oder hohen Ratio-Anforderungen out-of-memory memory-to-CPU

R.2X — Speicheroptimiertes Medium*

- DPU: 2 M-DPU (8 V, 64 GB Speicher) CPUs
- Anwendungsfall: Speicherintensive Workloads mit häufigen Fehlern oder hohen Ratio-Anforderungen out-of-memory memory-to-CPU

R.4X — Speicheroptimiert groß*

- DPU: 4 M-DPU (16 V, 128 GB Speicher) CPUs
- Anwendungsfall: Große speicherintensive Workloads mit häufigen Fehlern oder hohen Verhältnisanforderungen out-of-memory memory-to-CPU

R.8X — Speicheroptimiert, extra groß*

- DPU: 8 M-DPU (32 V, 256 GB Speicher) CPUs

- Anwendungsfall: Sehr große speicherintensive Workloads mit häufigen Fehlern oder hohen Verhältnisanforderungen out-of-memory memory-to-CPU

* Bei diesen Workern kann es zu einer höheren Startlatenz kommen. Versuchen Sie, das Problem wie folgt zu beheben:

- Warten Sie ein paar Minuten und reichen Sie Ihren Job dann erneut ein.
- Reichen Sie einen neuen Job mit einer reduzierten Anzahl von Mitarbeitern ein.
- Reichen Sie einen neuen Job mit einem anderen Arbeitnehmertyp oder einer anderen Mitarbeitergröße ein.

Tabelle mit Spezifikationen für den Arbeitnehmertyp

Spezifikationen für den Arbeitnehmertyp

Art des Arbeitnehmers	DPU pro Knoten	vCPU	Speicher (GB)	Festplatte (GB)	Freier Festplattenspeicher (GB)	Spark-Executoren pro Knoten
G.1X	1	4	16	94	44	1
G.2X	2	8	32	138	78	1
G.4X	4	16	64	256	230	1
G.8X	8	32	128	512	485	1
G12X	12	48	192	768	741	1
G.16X	16	64	256	1024	996	1

Hinweis: R-Worker-Typen verfügen über speicheroptimierte Konfigurationen mit Spezifikationen, die für speicherintensive Workloads optimiert sind.

Wichtige Überlegungen

Latenz beim Start

> **Important**
>
> Bei den Workertypen G.12X und G.16X sowie bei allen R-Workertypen (R.1X bis R.8X) kann es zu einer höheren Startlatenz kommen. Versuchen Sie, das Problem wie folgt zu beheben:
>
> - Warten Sie ein paar Minuten und reichen Sie Ihren Job dann erneut ein.
> - Reichen Sie einen neuen Job mit einer reduzierten Anzahl von Mitarbeitern ein.
> - Reichen Sie einen neuen Job mit einem anderen Mitarbeitertyp und einer anderen Mitarbeitergröße ein.

Auswahl des richtigen Arbeitnehmertyps

Für Standard-ETL-Workloads

- G.1X oder G.2X: Am kostengünstigsten für typische Datentransformationen, Verknüpfungen und Abfragen
- G.4X oder G.8X: Für anspruchsvollere Workloads mit größeren Datensätzen

Für umfangreiche Workloads

- G.12X: Sehr große Datensätze, die erhebliche Rechenressourcen erfordern
- G.16X: Maximale Rechenkapazität für die anspruchsvollsten Workloads

Für speicherintensive Workloads

- R.1X oder R.2X: Kleine bis mittlere speicherintensive Jobs
- R.4X oder R.8X: Große speicherintensive Workloads mit häufigen OOM-Fehlern

Überlegungen zur Kostenoptimierung

- Standard G-Worker: Stellen ein ausgewogenes Verhältnis von Rechen-, Arbeitsspeicher- und Netzwerkressourcen bereit und können für eine Vielzahl unterschiedlicher Workloads zu geringeren Kosten eingesetzt werden
- R-Worker: Spezialisiert auf speicherintensive Aufgaben mit schneller Leistung für Workloads, die große Datenmengen im Speicher verarbeiten

Bewährte Methoden

Richtlinien für die Auswahl von Arbeitnehmern

1. Beginnen Sie für die meisten Workloads mit Standardarbeitern (G.1X, G.2X)
2. Verwenden Sie R-Worker bei häufigen out-of-memory Fehlern oder Workloads mit speicherintensiven Vorgängen wie Zwischenspeichern, Mischen und Aggregieren
3. Ziehen Sie G.12X/G.16X für rechenintensive Workloads in Betracht, die maximale Ressourcen erfordern
4. Berücksichtigen Sie Kapazitätsengpässe, wenn Sie neue Mitarbeitertypen in zeitkritischen Workflows verwenden

Leistungsoptimierung

- Überwachen Sie CloudWatch Kennzahlen, um die Ressourcennutzung besser zu verstehen
- Verwenden Sie je nach Datengröße und Komplexität eine angemessene Anzahl von Mitarbeitern
- Erwägen Sie Strategien zur Datenpartitionierung, um die Effizienz der Mitarbeiter zu optimieren

ETL-Jobs streamen AWS Glue

Sie können Streaming-Aufträge für ETL-Aufträge (Extract, Transform, Load) erstellen, die kontinuierlich ausgeführt werden und Daten aus Streaming-Quellen wie Amazon Kinesis Data Streams, Apache Kafka und Amazon Managed Streaming for Apache Kafka (Amazon MSK) konsumieren. Die Aufträge bereinigen und transformieren die Daten und laden die Ergebnisse dann in Amazon-S3-Data-Lakes oder JDBC-Datenspeicher.

Darüber hinaus können Sie Daten zu Amazon Kinesis Data Streams erstellen. Diese Funktion ist nur beim Schreiben von AWS Glue Skripten verfügbar. Weitere Informationen finden Sie unter the section called “Kinesis-Verbindungen”.

Standardmäßig AWS Glue verarbeitet und schreibt Daten in 100-Sekunden-Fenstern aus. Dadurch können Daten effizient verarbeitet und Aggregationen für Daten ausgeführt werden, die später als erwartet eintreffen. Sie können die Größe dieses Zeitfensters ändern, um die Aktualität oder Aggregationsgenauigkeit zu erhöhen. AWS Glue Streaming-Jobs verwenden Checkpoints statt Job-Lesezeichen, um die gelesenen Daten nachzuverfolgen.

> **Note**
>
> AWS Glue berechnet stündlich das Streamen von ETL-Jobs, während sie ausgeführt werden.

In diesem Video werden die Kostenprobleme beim Streamen von ETL und die Funktionen zur Kosteneinsparung in AWS Glue beschrieben.

Die Erstellung eines Streaming-ETL-Auftrags umfasst die folgenden Schritte:

1. Erstellen Sie für eine Apache Kafka-Streaming-Quelle eine AWS Glue Verbindung zur Kafka-Quelle oder zum Amazon MSK-Cluster.
2. Erstellen Sie manuell eine Data-Catalog-Tabelle für die Streaming-Quelle.
3. Erstellen Sie einen ETL-Auftrag für die Streaming-Datenquelle. Definieren Sie streamingspezifische Auftragseigenschaften und geben Sie Ihr eigenes Skript an oder ändern Sie optional das generierte Skript.

Weitere Informationen finden Sie unter ETL einstreamen AWS Glue.

Wenn Sie einen Streaming-ETL-Job für Amazon Kinesis Data Streams erstellen, müssen Sie kein AWS Glue Verbindung. Wenn jedoch eine Verbindung mit dem verbunden ist AWS Glue Streaming-ETL-Job mit Kinesis Data Streams als Quelle, dann ist ein Virtual Private Cloud (VPC) -Endpunkt für Kinesis erforderlich. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Benutzerhandbuch für Amazon VPC. Wenn Sie einen Amazon-Kinesis-Data-Streams-Stream in einem anderen Konto angeben, müssen Sie die Rollen und Richtlinien einrichten, um den kontenübergreifenden Zugriff zu ermöglichen. Weitere Informationen finden Sie unter Beispiel: Aus einem Kinesis Stream in einem anderen Konto lesen.

AWS Glue Streaming-ETL-Jobs können komprimierte Daten automatisch erkennen, die Streaming-Daten transparent dekomprimieren, die üblichen Transformationen an der Eingabequelle durchführen und in den Ausgabespeicher laden.

AWS Glue unterstützt die automatische Dekomprimierung für die folgenden Komprimierungstypen je nach Eingabeformat:

Komprimierungsart	Avro-Datei	Avro-Datum	JSON	CSV	Grok
BZIP2	Ja	Ja	Ja	Ja	Ja
GZIP	Nein	Ja	Ja	Ja	Ja
SNAPPY	Ja (rohes Snappy)	Ja (gerahmtes Snappy)	Ja (gerahmtes Snappy)	Ja (gerahmtes Snappy)	Ja (gerahmtes Snappy)
XZ	Ja	Ja	Ja	Ja	Ja
ZSTD	Ja	Nein	Nein	Nein	Nein
DEFLATE	Ja	Ja	Ja	Ja	Ja

Themen

- Erstellen eines AWS Glue Verbindung für einen Apache Kafka-Datenstrom
- Erstellen einer Data-Catalog-Tabelle für eine Streaming-Quelle
- Hinweise und Einschränkungen für Avro-Streaming-Quellen
- Anwenden von Grok-Mustern auf Streaming-Quellen
- Definieren von Auftragseigenschaften für einen Streaming-ETL-Auftrag
- Hinweise zu und Einschränkungen für Streaming-ETL

Erstellen eines AWS Glue Verbindung für einen Apache Kafka-Datenstrom

Um aus einem Apache Kafka-Stream zu lesen, müssen Sie einen erstellen AWS Glue Verbindung.

Um eine zu erstellen AWS Glue Verbindung für eine Kafka-Quelle (Konsole)

1. Öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/
2. Wählen Sie im Navigationsbereich unter Data catalog die Option Connections (Verbindungen) aus.
3. Wählen Sie Add connection (Verbindung hinzufügen) und geben Sie auf der Seite Set up your connection's properties (Einrichten der Verbindungseigenschaften) einen Verbindungsnamen ein.

   > **Note**
   >
   > Weitere Informationen zum Angeben von Verbindungseigenschaften finden Sie unter Eigenschaften der AWS Glue -Verbindung.

4. Wählen Sie für Verbindungstyp den Eintrag Kafka.
5. Geben Sie für Kafka-Bootstrap-Server URLs den Host und die Portnummer für die Bootstrap-Broker für Ihren Amazon MSK-Cluster oder Apache Kafka-Cluster ein. Verwenden Sie nur TLS-Endpunkte (Transport Layer Security (TLS)) zum Herstellen der ersten Verbindung mit dem Kafka-Cluster. Nur–Text-Endpunkte werden nicht unterstützt.

   Im Folgenden finden Sie eine Beispielliste mit Host-Port-Nummerpaaren für einen Amazon-MSK-Cluster.

   ```
   myserver1.kafka.us-east-1.amazonaws.com:9094,myserver2.kafka.us-
   east-1.amazonaws.com:9094,
   myserver3.kafka.us-east-1.amazonaws.com:9094
   ```

   Weitere Informationen zum Abrufen der Bootstrap-Broker-Informationen finden Sie unter Abrufen der Bootstrap Broker für einen Amazon-MSK-Cluster im Amazon Managed Streaming for Apache-Kafka-Entwicklerhandbuch.
6. Wenn Sie eine sichere Verbindung zur Kafka-Datenquelle wünschen, wählen Sie SSL-Verbindung erforderlich und geben für Standort des privaten CA-Zertifikats von Kafka einen gültigen Amazon S3 Pfad zu einem benutzerdefinierten SSL-Zertifikat ein.

   Für eine SSL-Verbindung zu selbstverwaltetem Kafka ist das benutzerdefinierte Zertifikat obligatorisch. Es ist optional für Amazon MSK.

Weitere Informationen zur Angabe eines benutzerdefinierten Zertifikats für Kafka finden Sie unter the section called “SSL-Verbindungseigenschaften”.

7. Verwenden Sie AWS Glue Studio oder die AWS CLI, um eine Kafka-Client-Authentifizierungsmethode anzugeben. Um darauf zuzugreifen, AWS Glue Studio wählen Sie im ETL-Menü im linken Navigationsbereich AWS Glueaus.

   Weitere Informationen über Kafka-Client-Authentifizierungsmethoden finden Sie unter AWS Glue Kafka-Verbindungseigenschaften für die Client-Authentifizierung .
8. Geben Sie optional eine Beschreibung ein und wählen Sie dann Next (Weiter).
9. Geben Sie für einen Amazon-MSK-Cluster die Virtual Private Cloud (VPC), das Subnetz und die Sicherheitsgruppe an. Für selbstverwaltetes Kafka sind die VPC-Informationen optional.
10. Klicken Sie auf Next (Weiter), um alle Verbindungseigenschaften zu überprüfen, und wählen Sie dann Finish (Abschließen).

Weitere Informationen zur AWS Glue Verbindungen, sieheHerstellen einer Verbindung zu Daten.

AWS Glue Kafka-Verbindungseigenschaften für die Client-Authentifizierung

SASL/GSSAPI (Kerberos)-Authentifizierung

Wenn Sie diese Authentifizierungsmethode wählen, können Sie Kerberos-Eigenschaften angeben.

Kerberos-Keytab

Wählen Sie den Speicherort der Keytab-Datei aus. Ein Keytab speichert Langzeitschlüssel für ein oder mehrere Prinzipale. Weitere Informationen finden Sie unter MIT-Kerberos-Dokumentation: Keytab.

Kerberos krb5.conf-Datei

Wählen Sie die krb5.conf-Datei aus. Dies enthält den Standardbereich (ein logisches Netzwerk, ähnlich einer Domain, das eine Gruppe von Systemen unter demselben KDC definiert) und den Standort des KDC-Servers. Weitere Informationen finden Sie in der MIT-Kerberos-Dokumentation: krb5.conf.

Kerberos-Prinzipal und Kerberos-Dienstname

Geben Sie den Kerberos-Prinzipal und den Dienstnamen ein. Weitere Informationen finden Sie unter MIT Kerberos-Dokumentation: Kerberos-Prinzipal.

SASL/SCRAM-SHA-512-Authentifizierung

Wenn Sie diese Authentifizierungsmethode wählen, können Sie Anmeldeinformationen zur Authentifizierung angeben.

AWS Secrets Manager

Suchen Sie im Suchfeld nach Ihrem Token, indem Sie den Namen oder ARN eingeben.

Benutzername und Passwort des Anbieters direkt

Suchen Sie im Suchfeld nach Ihrem Token, indem Sie den Namen oder ARN eingeben.

SSL-Client-Authentifizierung

Wenn Sie diese Authentifizierungsmethode wählen, können Sie den Standort des Kafka-Client-Keystores auswählen, indem Sie Amazon S3 durchsuchen. Optional können Sie das Kennwort für den Kafka-Client-Keystore und das Kafka-Client-Schlüsselkennwort eingeben.

IAM-Authentifizierung

Diese Authentifizierungsmethode erfordert keine zusätzlichen Spezifikationen und ist nur anwendbar, wenn die Streaming-Quelle MSK Kafka ist.

SASL/PLAIN-Authentifizierung

Wenn Sie diese Authentifizierungsmethode wählen, können Sie Authentifizierungsdaten angeben.

Erstellen einer Data-Catalog-Tabelle für eine Streaming-Quelle

Eine Datenkatalogtabelle, die die Eigenschaften des Quelldatenstroms, einschließlich des Datenschemas, angibt, kann manuell für eine Streaming-Quelle erstellt werden. Diese Tabelle wird als Datenquelle für den Streaming-ETL-Auftrag verwendet.

Wenn Sie das Schema der Daten im Quelldatenstrom nicht kennen, können Sie die Tabelle ohne Schema erstellen. Wenn Sie dann den Streaming-ETL-Job erstellen, können Sie den AWS Glue Funktion zur Schemaerkennung. AWS Glue bestimmt das Schema anhand der Streaming-Daten.

Verwenden der AWS Glue Konsole, die AWS Command Line Interface (AWS CLI) oder die AWS Glue API zum Erstellen der Tabelle. Informationen zum manuellen Erstellen einer Tabelle mit dem AWS Glue Konsole finden Sie unterthe section called “Erstellen von Tabellen”.

> **Note**
>
> Sie können die AWS Lake Formation Konsole nicht verwenden, um die Tabelle zu erstellen. Sie müssen die AWS Glue console.

Beachten Sie auch die folgenden Informationen für Streaming-Quellen im Avro-Format oder für Protokolldaten, auf die Sie Grok-Muster anwenden können.

- the section called “Hinweise und Einschränkungen für Avro-Streaming-Quellen”
- the section called “Anwenden von Grok-Mustern auf Streaming-Quellen”

Themen

- Kinesis-Datenquelle
- Kafka-Datenquelle
- AWS Glue Quelle der Schemaregistrierungstabelle

Kinesis-Datenquelle

Legen Sie beim Erstellen der Tabelle die folgenden Streaming-ETL-Eigenschaften fest (Konsole).

Quelltyp

Kinesis

Für eine Kinesis-Quelle im selben Konto:

Region

Die AWS Region, in der sich der Amazon Kinesis Data Streams Streams-Service befindet. Der Name der Region und des Kinesis-Streams werden zusammen in einen Stream-ARN übersetzt.

Beispiel: https://kinesis.us-east-1.amazonaws.com

Kinesis-Streamname

Der Stream-Name wie unter Erstellen eines Streams im Entwicklerhandbuch zu Amazon Kinesis Data Streams beschrieben.

Informationen zu einer Kinesis-Quelle in einem anderen Konto finden Sie in diesem Beispiel zum Einrichten der Rollen und Richtlinien, um den kontenübergreifenden Zugriff zu ermöglichen. Konfigurieren Sie diese Einstellungen:

Stream-ARN

Der ARN des Kinesis-Datenstroms, mit dem der Verbraucher registriert ist. Weitere Informationen finden Sie unter Amazon Resource Names (ARNs) und AWS Service Namespaces in der. Allgemeine AWS-Referenz

Angenommene ARN-Rolle

Der angenommene Amazon-Ressourcenname (ARN) der Rolle.

Sitzungsname (optional)

Ein Bezeichner für die Sitzung der angenommenen Rolle.

Verwenden Sie den Namen der Rollensitzung, um eine Sitzung eindeutig zu identifizieren, wenn dieselbe Rolle von verschiedenen Prinzipalen oder aus unterschiedlichen Gründen übernommen wird. In kontenübergreifenden Szenarien ist der Name der Rollensitzung für das Konto sichtbar und kann von dem Konto protokolliert werden, dem die Rolle gehört. Der Rollensitzungsname wird auch im ARN des übernommenen Rollenprinzipals verwendet. Das bedeutet, dass nachfolgende kontoübergreifende API-Anfragen, die die temporären Sicherheitsanmeldedaten verwenden, den Namen der Rollensitzung für das externe Konto in ihren Protokollen offenlegen. AWS CloudTrail

Um Streaming-ETL-Eigenschaften für Amazon Kinesis Data Streams festzulegen (AWS Glue API oder AWS CLI)

- Um Streaming-ETL-Eigenschaften für eine Kinesis-Quelle in demselben Konto einzurichten, geben Sie die `streamName`- und `endpointUrl`-Parameter in der `StorageDescriptor`-Struktur der `CreateTable`-API-Operation oder dem `create_table`-CLI-Befehl an.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamName": "sample-stream",
  "endpointUrl": "https://kinesis.us-east-1.amazonaws.com"
 }
 ...
}
```

Oder geben Sie den `streamARN` an.

Example

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream"
 }
 ...
}
```

- Um Streaming-ETL-Eigenschaften für eine Kinesis-Quelle in einem anderen Konto einzurichten, geben Sie die `streamARN`-, `awsSTSRoleARN`- und (optional) `awsSTSSessionName`-Parameter in der `StorageDescriptor`-Struktur der `CreateTable`-API-Operation oder dem `create_table`-CLI-Befehl an.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream",
  "awsSTSRoleARN": "arn:aws:iam::123456789:role/sample-assume-role-arn",
  "awsSTSSessionName": "optional-session"
 }
 ...
}
```

Kafka-Datenquelle

Legen Sie beim Erstellen der Tabelle die folgenden Streaming-ETL-Eigenschaften fest (Konsole).

Quelltyp

Kafka

Für eine Kafka-Quelle:

Themenname

Der Topic-Name wie in Kafka angegeben.

Verbindung

Importieren in &S3; AWS Glue Verbindung, die auf eine Kafka-Quelle verweist, wie unter beschriebenthe section called “Erstellen einer Verbindung für einen Kafka-Datenstrom”.

AWS Glue Quelle der Schemaregistrierungstabelle

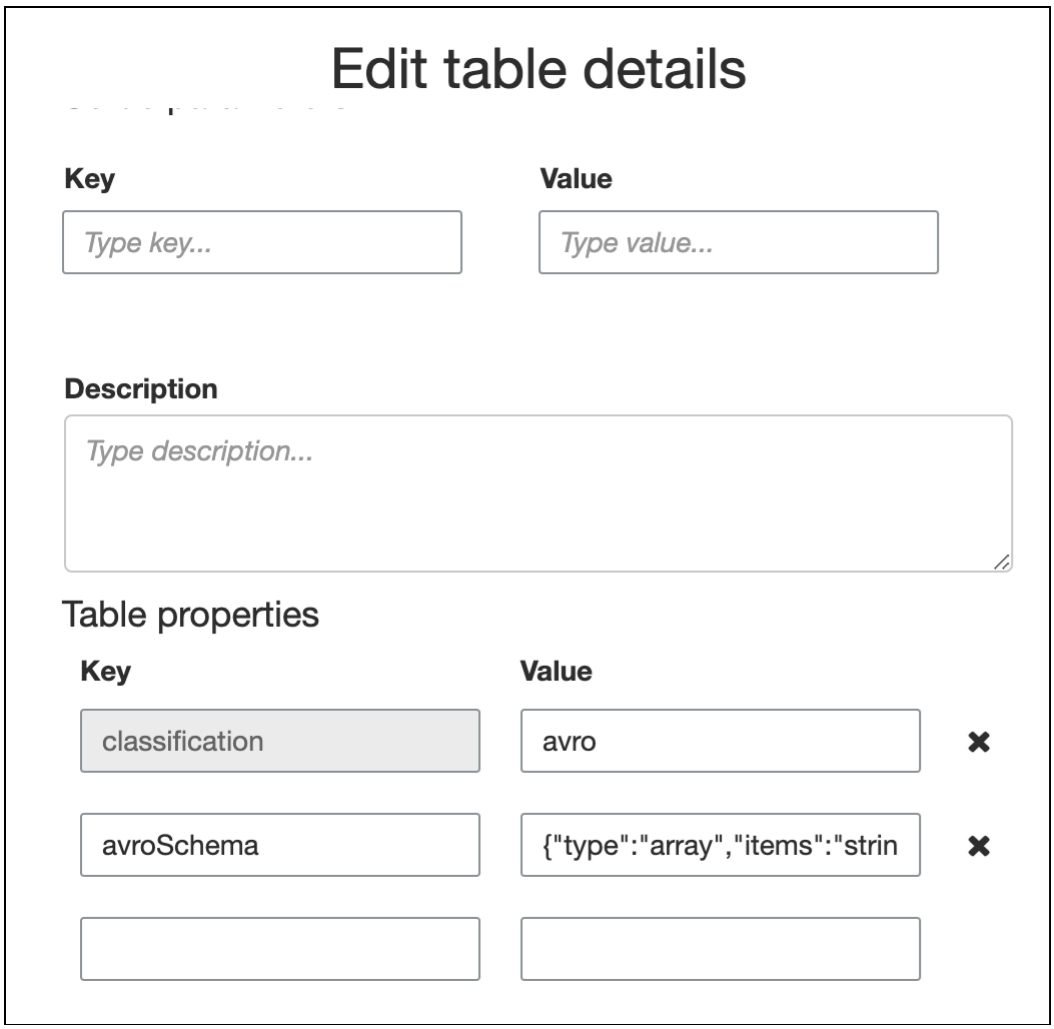
Zur Verwendung AWS Glue Schemaregistrierung für Streaming-Jobs: Folgen Sie den Anweisungen unter Anwendungsfall: AWS Glue Data Catalog So erstellen oder aktualisieren Sie eine Schemaregistry-Tabelle.

Derzeit AWS Glue Streaming unterstützt nur das Avro-Format von Glue Schema Registry mit der Einstellung Schemainferenz auf. `false`

Hinweise und Einschränkungen für Avro-Streaming-Quellen

Die folgenden Hinweise und Einschränkungen gelten für Streaming-Quellen im Avro-Format:

- Wenn die Schemaerkennung aktiviert ist, muss das Avro-Schema in die Nutzlast einbezogen werden. Wenn diese Option deaktiviert ist, sollte die Nutzlast nur Daten enthalten.
- Einige Avro-Datentypen werden in Dynamic Frames nicht unterstützt. Sie können diese Datentypen nicht angeben, wenn Sie das Schema auf der Seite Schema definieren im Assistenten zur Tabellenerstellung im AWS Glue console. Während der Schemaerkennung werden nicht unterstützte Typen im Avro-Schema wie folgt in unterstützte Typen konvertiert:
  - `EnumType => StringType`
  - `FixedType => BinaryType`
  - `UnionType => StructType`
- Wenn Sie das Tabellenschema auf der Seite Definieren eines Schemas in der Konsole definieren, ist der implizierte Stammelementtyp für das Schema `record`. Wenn Sie einen anderen Stammelementtyp als `record` möchten, zum Beispiel `array` oder `map`, können Sie das Schema nicht mithilfe der Seite Definieren eines Schemas angeben. Stattdessen müssen Sie diese Seite überspringen und das Schema entweder als Tabelleneigenschaft oder im ETL-Skript angeben.
  - Um das Schema in den Tabelleneigenschaften anzugeben, füllen Sie den Assistenten zum Erstellen von Tabellen aus, bearbeiten Sie die Tabellendetails und fügen Sie unter Tabelleneigenschaften ein neues Schlüssel-Wert-Paar hinzu. Verwenden Sie den Schlüssel `avroSchema` und geben Sie ein Schema-JSON-Objekt für den Wert ein, wie im folgenden Screenshot gezeigt.



- Um das Schema im ETL-Skript anzugeben, ändern Sie die `datasource0`-Zuweisungsanweisung und fügen Sie den Schlüssel `avroSchema` zum Argument `additional_options` hinzu, wie in den folgenden Python- und Scala-Beispielen gezeigt.

  Python

  ```
  SCHEMA_STRING = ‘{"type":"array","items":"string"}’
  datasource0 = glueContext.create_data_frame.from_catalog(database =
   "database", table_name = "table_name", transformation_ctx = "datasource0",
   additional_options = {"startingPosition": "TRIM_HORIZON", "inferSchema":
   "false", "avroSchema": SCHEMA_STRING})
  ```

  Scala

  ```
  val SCHEMA_STRING = """{"type":"array","items":"string"}"""
  val datasource0 = glueContext.getCatalogSource(database = "database", tableName
   = "table_name", redshiftTmpDir = "", transformationContext = "datasource0",
  ```

```
additionalOptions = JsonOptions(s"""{"startingPosition": "TRIM_HORIZON",
"inferSchema": "false", "avroSchema":"$SCHEMA_STRING"}""")).getDataFrame()
```

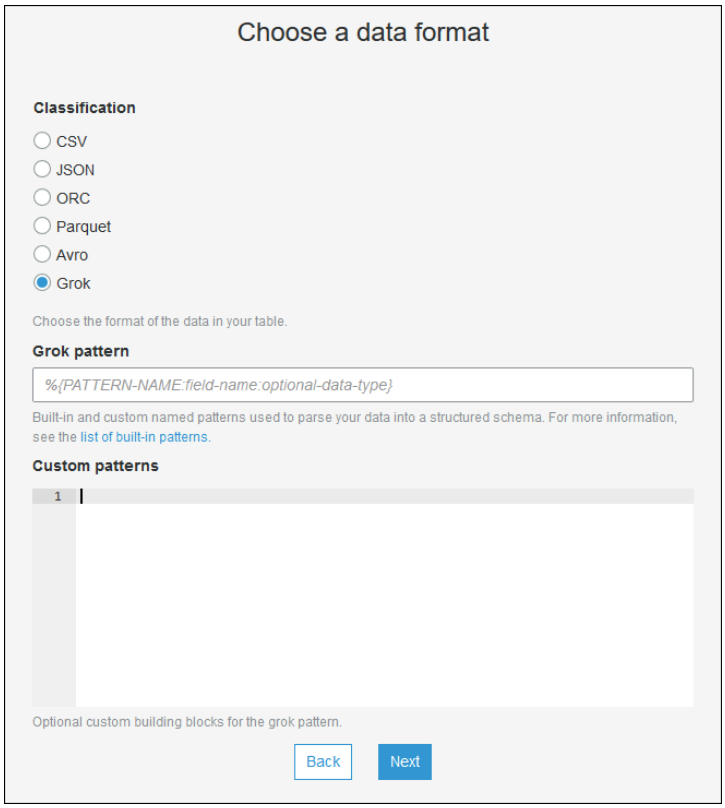
Anwenden von Grok-Mustern auf Streaming-Quellen

Sie können einen Streaming-ETL-Auftrag für eine Protokolldatenquelle erstellen und Grok-Muster verwenden, um die Protokolle in strukturierte Daten zu konvertieren. Der ETL-Auftrag verarbeitet die Daten dann als strukturierte Datenquelle. Sie geben die Grok-Muster an, die angewendet werden sollen, wenn Sie die Data-Catalog-Tabelle für die Streaming-Quelle erstellen.

Informationen zu Grok-Mustern und benutzerdefinierten Musterzeichenfolgenwerten finden Sie unter Angepasste Grok-Classifier schreiben.

So fügen Sie der Data-Catalog-Tabelle (Konsole) Grok-Muster hinzu

- Verwenden Sie den Assistenten zum Erstellen von Tabellen, und erstellen Sie die Tabelle mit den in the section called “Erstellen einer Data-Catalog-Tabelle für eine Streaming-Quelle” angegebenen Parametern. Geben Sie als Datenformat Grok an, füllen Sie das Feld Grok-Muster aus und fügen Sie optional benutzerdefinierte Muster unter Benutzerdefinierte Muster (optional) hinzu.



Drücken Sie nach jedem benutzerdefinierten Muster auf Eingabe.

Um Grok-Muster zur Datenkatalogtabelle hinzuzufügen (AWS Glue API oder AWS CLI)

- Fügen Sie den `GrokPattern`-Parameter und optional den `CustomPatterns`-Parameter zu der `CreateTable`-API-Operation oder dem `create_table`-CLI-Befehl hinzu.

```
 "Parameters": {
...
    "grokPattern": "string",
    "grokCustomPatterns": "string",
...
},
```

Drücken Sie `grokCustomPatterns` als String aus und verwenden Sie „\ n“ als Trennzeichen zwischen Mustern.

Nachfolgend finden Sie ein Beispiel für die Angabe dieser Parameter.

Example

```
"parameters": {
...
    "grokPattern": "%{USERNAME:username} %{DIGIT:digit:int}",
    "grokCustomPatterns": "digit \d",
...
}
```

Definieren von Auftragseigenschaften für einen Streaming-ETL-Auftrag

Wenn Sie einen Streaming-ETL-Job in der definieren AWS Glue Geben Sie in der Konsole die folgenden streamspezifischen Eigenschaften an. Beschreibungen weiterer Auftragseigenschaften finden Sie unter Definieren von Auftragseigenschaften für Spark-Aufträge.

IAM-Rolle

Geben Sie die AWS Identity and Access Management (IAM-) Rolle an, die für die Autorisierung von Ressourcen verwendet wird, die für die Ausführung des Jobs, den Zugriff auf Streaming-Quellen und den Zugriff auf Zieldatenspeicher verwendet werden.

Für den Zugriff auf Amazon Kinesis Data Streams hängen Sie die `AmazonKinesisFullAccess` AWS verwaltete Richtlinie an die Rolle an oder fügen Sie eine ähnliche IAM-Richtlinie hinzu, die einen detaillierteren Zugriff ermöglicht. Beispielrichtlinien finden Sie unter Steuern des Zugriffs auf Amazon Kinesis Data Streams-Ressourcen mithilfe von IAM.

Weitere Informationen zu Berechtigungen für die Ausführung von Jobs finden Sie in AWS Glue, finden Sie unter Identitäts- und Zugriffsmanagement für AWS Glue.

Typ

Wählen Sie Spark streaming (Spark-Streaming).

AWS Glue version

Das Tool AWS Glue Version bestimmt die Versionen von Apache Spark und Python oder Scala, die für den Job verfügbar sind. Wählen Sie eine Auswahl, die die für den Job verfügbare Version von Python oder Scala angibt. AWS Glue Version 2.0 mit Python-3-Unterstützung ist die Standardeinstellung für Streaming-ETL-Aufträge.

Wartungsfenster

Gibt ein Fenster an, in dem ein Streaming-Job neu gestartet werden kann. Siehe the section called “Wartungsfenster”.

Zeitüberschreitung von Aufträgen

Geben Sie optional eine Dauer in Minuten ein. Der Standardwert ist leer.

- Streaming-Jobs müssen einen Timeout-Wert von weniger als 7 Tagen oder 10080 Minuten haben.
- Wenn der Wert leer gelassen wird, wird der Job nach 7 Tagen neu gestartet, sofern Sie kein Wartungsfenster eingerichtet haben. Wenn Sie ein Wartungsfenster eingerichtet haben, wird der Job während des Wartungsfensters nach 7 Tagen neu gestartet.

Datenquelle

Geben Sie die Tabelle an, die Sie in the section called “Erstellen einer Data-Catalog-Tabelle für eine Streaming-Quelle” erstellt haben.

Datenziel

Führen Sie eine der folgenden Aktionen aus:

- Wählen Sie Create tables in your data target (Tabellen in eigenem Datenziel erstellen) und geben Sie die folgenden Eigenschaften für das Datenziel an.

Datastore

Wählen Sie Amazon S3 oder JDBC.

Format

Wählen Sie ein beliebiges Format aus. Für das Streaming werden alle unterstützt.

- Wählen Sie Use tables in the data catalog and update your data target (Tabellen im Data Catalog verwenden und Datenziel aktualisieren) und wählen Sie eine Tabelle für einen JDBC-Datastore.

Ausgabeschemadefinition

Führen Sie eine der folgenden Aktionen aus:

- Wählen Sie Schema für jeden Datensatz automatisch erkennen, um die Schemaerkennung zu aktivieren. AWS Glue bestimmt das Schema anhand der Streaming-Daten.
- Klicken Sie auf Specify output schema for all records (Ausgabeschema für alle Datensätze angeben), um das Ausgabeschema mithilfe der Transformation „Apply Mapping“ (Mapping anwenden) zu definieren.

Script

Geben Sie optional Ihr eigenes Skript an oder ändern Sie das generierte Skript, um Operationen auszuführen, die von der Apache-Spark-Engine Structured Streaming unterstützt werden. Informationen zu den verfügbaren Vorgängen finden Sie unter Operationen beim Streamen DataFrames /Datasets.

Hinweise zu und Einschränkungen für Streaming-ETL

Beachten Sie die folgenden Hinweise und Einschränkungen:

- Automatische Dekomprimierung für AWS Glue Das Streamen von ETL-Jobs ist nur für die unterstützten Komprimierungstypen verfügbar. Beachten Sie auch das Folgende:
  - Das gerahmte Snappy bezieht sich auf das offizielle Rahmen-Format für Snappy.
  - Deflate wird in der Glue-Version 3.0 unterstützt, nicht in der Glue-Version 2.0.
- Wenn Sie die Schemaerkennung verwenden, können Sie keine Joins von Streamingdaten ausführen.
- AWS Glue Streaming-ETL-Jobs unterstützen nicht den Union-Datentyp für AWS Glue Schemaregistrierung im Avro-Format.

- Ihr ETL-Skript kann verwenden AWS GlueDie integrierten Transformationen und die systemeigenen Transformationen von Apache Spark Structured Streaming. Weitere Informationen finden Sie unter Operationen beim Streamen DataFrames /Datasets auf der Apache Spark-Website oder. AWS Glue PySpark transformiert Referenz
- AWS Glue Streaming-ETL-Jobs verwenden Checkpoints, um den Überblick über die gelesenen Daten zu behalten. Daher wird ein Auftrag, der angehalten und neu gestartet wurde, an dem Punkt fortgesetzt, an dem er im Stream beendet wurde. Wenn Sie Daten neu verarbeiten möchten, können Sie den Checkpoint-Ordner löschen, auf den im Skript verwiesen wird.
- Auftragslesezeichen werden nicht unterstützt.
- Zur Verwendung des erweiterten Fan-Out-Features von Kinesis Data Streams in Ihrem Auftrag lesen Sie the section called “Verwendung von erweitertem Fan-Out in Kinesis-Streaming-Aufträgen”.
- Wenn Sie eine Datenkatalogtabelle verwenden, die aus erstellt wurde AWS Glue Wenn in der Schemaregistrierung eine neue Schemaversion verfügbar ist, müssen Sie wie folgt vorgehen, um das neue Schema widerzuspiegeln:
  1. Stoppen Sie die mit der Tabelle verknüpften Aufträge.
  2. Aktualisieren Sie das Schema für die Data Catalog-Tabelle.
  3. Starten Sie die mit der Tabelle verknüpften Aufträge neu.

Datensatzabgleich mit AWS Lake Formation FindMatches

> **Note**
>
> Der Datensatzabgleich ist derzeit in den folgenden Regionen der AWS Glue Konsole nicht verfügbar: Naher Osten (VAE), Europa (Spanien), Asien-Pazifik (Jakarta) und Europa (Zürich).

AWS Lake Formation bietet Funktionen für maschinelles Lernen, mit denen Sie benutzerdefinierte Transformationen zur Bereinigung Ihrer Daten erstellen können. Derzeit ist eine Transformation mit dem Namen verfügbar. FindMatches Die FindMatches Transformation ermöglicht es Ihnen, doppelte oder übereinstimmende Datensätze in Ihrem Datensatz zu identifizieren, selbst wenn die Datensätze keinen gemeinsamen eindeutigen Bezeichner haben und keine Felder exakt übereinstimmen. Dazu müssen Sie keinen Code schreiben oder wissen, wie maschinelles Lernen funktioniert. FindMatches kann bei vielen verschiedenen Problemen nützlich sein, wie zum Beispiel:

- Abgleichen von Kunden: Verknüpfen von Kundendatensätzen über verschiedene Kundendatenbanken hinweg, auch wenn viele Kundenfelder in den Datenbanken nicht exakt übereinstimmen (z. B. unterschiedliche Schreibweise der Namen, Adressunterschiede, fehlende oder ungenaue Daten usw.).
- Abgleichen von Produkten: Abgleichen von Produkten in Ihrem Katalog mit anderen Produktquellen, beispielsweise des Produktkatalogs mit dem Katalog eines Wettbewerbers, wobei Einträge unterschiedlich strukturiert sind.
- Verbesserung der Betrugserkennung: Identifizieren von doppelten Kundenkonten, um zu bestimmen, wann ein neu erstelltes Konto mit einem zuvor bekannten betrügerischen Benutzer übereinstimmt (oder übereinstimmen könnte).
- Andere Übereinstimmungsprobleme: Übereinstimmung von Adressen, Filmen, Teilelisten usw. Wenn ein Mensch sich Ihre Datenbankzeilen ansehen und feststellen könnte, dass sie übereinstimmen, besteht im Allgemeinen eine sehr gute Chance, dass die FindMatches Transformation Ihnen helfen kann.

Sie können diese Transformationen erstellen, wenn Sie einen Auftrag erstellen. Die von Ihnen erstellte Transformation basiert auf einem Quelldatenspeicherschema und Beispieldaten aus dem Quelldatensatz, denen Sie Labels zuweisen (wir bezeichnen diesen Prozess als „Trainieren" einer Transformation). Die von Ihnen mit einem Label versehenen Datensätze müssen im Quelldatensatz vorhanden sein. In diesem Prozess generieren wir eine Datei, die Sie labeln und dann wieder hochladen, woraus die Transformation in gewisser Weise lernen würde. Nachdem Sie Ihre Transformation gelehrt haben, können Sie sie von Ihrer Spark-Plattform aus aufrufen AWS Glue Job (PySpark oder Scala Spark) und ihn in anderen Skripten mit einem kompatiblen Quelldatenspeicher verwenden.

Nachdem die Transformation erstellt wurde, wird sie gespeichert in AWS Glue. Auf der AWS Glue In der Konsole können Sie die von Ihnen erstellten Transformationen verwalten. Im Navigationsbereich unter Datenintegration und ETL, Datenklassifizierungs-Tools > Datensatzabgleich, können Sie Ihre Machine-Learning-Transformation bearbeiten und weiter unterrichten. Weitere Informationen zum Verwalten von Transformationen in der Konsole finden Sie unter Die Arbeit mit maschinellem Lernen transformiert.

> **Note**
>
> AWS Glue FindMatches Jobs der Version 2.0 verwenden den Amazon S3 S3-Bucket`aws-glue-temp-<accountID>-<region>`, um temporäre Dateien zu speichern, während die Transformation Daten verarbeitet. Sie können diese Daten, nachdem die Ausführung

abgeschlossen wurde, entweder manuell oder durch Festlegen einer Amazon S3-Lebenszyklusregel löschen.

Arten von Machine Learning-Transformationen

Sie können Machine Learning-Transformationen erstellen, um Ihre Daten zu bereinigen. Sie können diese Transformationen aus Ihrem ETL-Skript heraus aufrufen. Ihre Daten werden von Transformation zu Transformation in einer Datenstruktur namens a DynamicFrameweitergeleitet. Dabei handelt es sich um eine Erweiterung von Apache Spark SQL`DataFrame`. Der `DynamicFrame` enthält Ihre Daten und Sie verweisen auf das Schema, um Ihre Daten zu verarbeiten.

Die folgenden Arten von Machine Learning-Transformationen sind verfügbar:

Suche nach Übereinstimmungen

Sucht doppelte Datensätze in den Quelldaten. Zum Trainieren dieser Machine Learning-Transformation kennzeichnen Sie durch entsprechendes Labeling von Beispieldatensätzen, welche Zeilen übereinstimmen. Die Machine Learning-Transformation lernt immer besser, welche Zeilen Übereinstimmungen darstellen sollten, je mehr Sie sie mit gelabelten Beispieldaten trainieren. Abhängig davon, wie Sie die Transformation konfigurieren, ist eine der folgenden Ausgaben möglich:

- Eine Kopie der Eingabetabelle sowie eine `match_id`-Spalte mit Werten, die übereinstimmende Gruppen von Datensätzen angeben. Die `match_id`-Spalte ist ein beliebiger Bezeichner. Alle Datensätze mit derselben `match_id` wurden als übereinstimmend identifiziert. Datensätze mit anderer `match_id`stimmen nicht überein.
- Eine Kopie der Eingabetabelle, in der doppelte Zeilen entfernt wurden. Wenn mehrere Duplikate gefunden werden, wird der Datensatz mit dem niedrigsten Primärschlüssel beibehalten.

Inkrementelle Übereinstimmungen finden

Die Transformation „Übereinstimmungen suchen“ kann auch konfiguriert werden, um Übereinstimmungen in den vorhandenen und inkrementellen Frames zu finden und als Ausgabe eine Spalte zurückzugeben, die eine eindeutige ID pro Übereinstimmungsgruppe enthält.

Weitere Informationen finden Sie unter: Inkrementelle Übereinstimmungen finden

Verwenden der FindMatches Transformation

Sie können mit der `FindMatches`-Transformation nach doppelten Datensätzen in den Quelldaten suchen. Zur Unterstützung beim Trainieren der Transformation wird eine Labeling-Datei generiert oder zur Verfügung gestellt.

> **Note**
>
> Derzeit werden `FindMatches`-Transformationen, die einen benutzerdefinierten Verschlüsselungsschlüssel verwenden, in den folgenden Regionen nicht unterstützt:
>
> - Asien Pazifik (Osaka) – `ap-northeast-3`

Um mit der FindMatches Transformation zu beginnen, können Sie die folgenden Schritte ausführen. Ein ausführlicheres und detaillierteres Beispiel finden Sie im AWS Big Data-Blog: Harmonize data using AWS Glue und AWS Lake Formation FindMatches ML, um eine 360-Grad-Ansicht für Kunden zu erstellen.

Erste Schritte mit der Transformation zur Suche nach Übereinstimmungen

Führen Sie als Einstieg in die `FindMatches`-Transformation die folgenden Schritte aus:

1. Erstellen Sie im eine Tabelle AWS Glue Data Catalog für die Quelldaten, die bereinigt werden sollen. Informationen zum Erstellen eines Crawlers finden Sie unter Arbeiten mit Crawlern auf der AWS Glue Konsole.

   Wenn es sich bei Ihren Quelldaten um eine textbasierte Datei handelt, wie z. B. eine CSV-Datei (durch Kommas voneinander getrennte Werte), berücksichtigen Sie Folgendes:

   - Bewahren Sie die CSV-Datei Ihres Eingabedatensatzes und die Labeling-Datei in separaten Ordnern auf. Andernfalls ist der AWS Glue Der Crawler betrachtet sie möglicherweise als mehrere Teile derselben Tabelle und erstellt Tabellen im Datenkatalog falsch.
   - Außer für CSV-Dateien, die ausschließlich ASCII-Zeichen enthalten, stellen Sie sicher, dass für die CSV-Dateien UTF-8 ohne BOM-Codierung (Byte Order Mark, Markierung der Bytereihenfolge) verwendet wird. Microsoft Excel fügt am Anfang der UTF-8-CSV-Dateien häufig eine BOM ein. Um diese zu entfernen, öffnen Sie die CSV-Datei in einem Texteditor und speichern Sie sie mit der Option UTF-8 without BOM (UTF-8 ohne BOM) neu.
2. Auf dem AWS Glue Erstellen Sie in der Konsole einen Job und wählen Sie den Transformationstyp Find Matches aus.

> **Important**
>
> Die Datenquellentabelle, die Sie für den Auftrag auswählen, darf nicht mehr als 100 Spalten enthalten.

3. Sag AWS Glue um eine Beschriftungsdatei zu generieren, indem Sie Beschriftungsdatei generieren wählen. AWS Glue führt den ersten Schritt beim Gruppieren ähnlicher Datensätze für jeden Datensatz durch, `labeling_set_id` sodass Sie diese Gruppierungen überprüfen können. Sie kennzeichnen Übereinstimmungen in der Spalte `label`.
   - Wenn Sie bereits über eine Labeling-Datei, also ein Beispiel von Datensätzen mit übereinstimmenden Zeilen, verfügen, laden Sie die Datei in Amazon Simple Storage Service (Amazon S3) hoch. Weitere Informationen über das Format der Labeling-Datei finden Sie unter Format der Labeling-Datei. Fahren Sie mit Schritt 4 fort.
4. Laden Sie die Labeling-Datei herunter und labeln Sie die Datei wie im Abschnitt Labeling beschrieben.
5. Laden Sie die korrigierte beschriftete Datei hoch. AWS Glue führt Aufgaben aus, um der Transformation das Finden von Übereinstimmungen beizubringen.

   Wählen Sie auf der Seite mit der Liste Machine learning transforms (Maschine Learning-Transformationen) die Registerkarte History (Verlauf) aus. Auf dieser Seite wird angegeben, wann AWS Glue führt die folgenden Aufgaben aus:
   - Import labels (Labels importieren)
   - Export labels (Labels exportieren)
   - Generate labels (Labels generieren)
   - Estimate quality (Qualität beurteilen)
6. Um eine bessere Transformation zu erstellen, können Sie die gelabelte Datei iterativ herunterladen, kennzeichnen und wieder hochladen. Bei den anfänglichen Ausführungen werden möglicherweise sehr viel mehr Datensätze falsch zugeordnet. Aber AWS Glue lernt, während Sie es weiter vermitteln, indem Sie die Kennzeichnungsdatei überprüfen.
7. Beurteilen und optimieren Sie Ihre Transformation, indem Sie die Leistung und Ergebnisse der Suche nach Übereinstimmungen beurteilen. Weitere Informationen finden Sie unter Die Optimierung des maschinellen Lernens verwandelt sich in AWS Glue.

Labeling

Wenn `FindMatches` eine Labeling-Datei generiert, werden Datensätze aus Ihrer Quelltabelle ausgewählt. `FindMatches` erkennt basierend auf vorherigem Training die wertvollsten Datensätze, von denen es lernen kann.

Unter Labeling ist das Bearbeiten einer Labeling-Datei (z. B. einer Tabellenkalkulation wie aus Microsoft Excel) und das Hinzufügen von Kennungen oder Labels zur Spalte `label` zu verstehen, in der übereinstimmende und nicht übereinstimmende Datensätze identifiziert werden. Es ist wichtig, dass in Ihren Quelldaten klar und konsistent definiert ist, woran eine Übereinstimmung zu erkennen ist. `FindMatches` lernt davon, welche Datensätze von Ihnen als Übereinstimmungen angesehen werden, sowie anhand Ihrer Entscheidungen, wie doppelte Datensätze zu finden sind.

Wenn eine Labeling-Datei von `FindMatches` geniert wird, werden ca. 100 Datensätze angelegt. Diese 100 Datensätze sind in der Regel in 10 Labeling-Sätze unterteilt, wobei jeder Labeling-Satz durch eine eindeutige `labeling_set_id` identifiziert wird, die von `FindMatches` generiert wird. Jeder Labeling-Satz sollte unabhängig von den anderen Labeling-Sätzen als separate Labeling-Aufgabe betrachtet werden. Ihre Aufgabe besteht darin, übereinstimmende und nicht übereinstimmende Datensätze innerhalb jedes Labeling-Satzes zu identifizieren.

Tipps zum Bearbeiten von Labeling-Dateien in einer Tabellenkalkulation

Beim Bearbeiten der Labeling-Datei in einer Tabellenkalkulationsanwendung sollten Sie Folgendes berücksichtigen:

- Die Datei wird möglicherweise nicht mit vollständig erweiterten Spaltenfeldern geöffnet. Sie müssen möglicherweise die Spalten `labeling_set_id` und `label` erweitern, um den Inhalt in diesen Zellen sichtbar zu machen.
- Wenn die Primärschlüsselspalte eine Zahl ist, z. B. ein `long`-Datentyp, legt die Kalkulationstabelle sie möglicherweise als Zahl aus und ändert den Wert. Dieser Schlüssewertl muss als Text behandelt werden. Um dieses Problem zu beheben, formatieren Sie alle Zellen in der Primärschlüsselspalte als Text data (Textdaten).

Format der Labeling-Datei

Die Beschriftungsdatei, die generiert wird von AWS Glue um Ihre `FindMatches` Transformation beizubringen, wird das folgende Format verwendet. Wenn Sie Ihre eigene Datei für generieren AWS Glue, sie muss ebenfalls diesem Format folgen:

- Sie ist eine CSV-Datei (durch Kommas voneinander getrennte Werte).
- Sie muss in `UTF-8` codiert sein. Wenn Sie die Datei mit Microsoft Windows bearbeiten, wird sie möglicherweise mit `cp1252` codiert.
- Es muss sich an einem Amazon S3-Standort befinden, an den es übergeben werden kann AWS Glue.
- Sie sollten eine moderate Anzahl an Zeilen für jede Labeling-Aufgabe verwenden. Es werden 10–20 Zeilen pro Aufgabe empfohlen, obwohl 2–30 Zeilen pro Aufgabe zulässig sind. Aufgaben, die größer als 50 Zeilen sind, werden nicht empfohlen und können zu schlechten Ergebnissen oder Systemausfällen führen.
- Wenn Sie bereits Daten mit Labeling haben, die aus Paaren von Datensätzen bestehen, die als „Übereinstimmung“ oder „Keine Übereinstimmung“ gekennzeichnet sind, ist dies in Ordnung. Diese Paare mit Labeling können als Labeling-Sätze der Größe 2 dargestellt werden. Bezeichnen Sie in diesem Fall beide Datensätze beispielsweise mit dem Buchstaben „A“, wenn sie übereinstimmen, aber bezeichnen Sie einen als „A“ und einen als „B“, wenn sie nicht übereinstimmen.

  > **Note**
  >
  > Aufgrund ihrer zusätzlichen Spalten weist die Labeling-Datei ein anderes Schema als die Datei mit den Quelldaten auf. Platzieren Sie die Beschriftungsdatei in einem anderen Ordner als jede CSV-Eingabedatei für die Transformation, sodass AWS Glue Crawler berücksichtigt dies nicht, wenn er Tabellen im Datenkatalog erstellt. Andernfalls werden die Tabellen, die von der erstellt wurden AWS Glue Der Crawler stellt Ihre Daten möglicherweise nicht korrekt dar.

- Die ersten beiden Spalten (`labeling_set_id`,`label`) werden benötigt von AWS Glue. Die verbleibenden Spalten müssen dem Schema der Daten entsprechen, die verarbeitet werden sollen.
- Sie identifizieren für jede `labeling_set_id` alle übereinstimmenden Datensätze unter Verwendung desselben Labels. Ein Label ist eine eindeutige Zeichenfolge, die in die Spalte `label` platziert wird. Wir raten zur Verwendung von Labels mit einfachen Zeichen, z. B. A, B, C usw. Labels unterscheiden zwischen Groß- und Kleinschreibung und werden in die Spalte `label` eingegeben.
- Zeilen, die dieselbe `labeling_set_id` und dasselbe Label enthalten, werden als Übereinstimmung bezeichnet.
- Zeilen, die dieselbe `labeling_set_id` und ein anderes Label enthalten, werden als nicht übereinstimmend bezeichnet.

- Zeilen, die eine andere `labeling_set_id` enthalten, werden so gewertet, dass sie keine Informationen für oder gegen eine Übereinstimmung enthalten.

  Nachfolgend finden Sie ein Beispiel für das Kennzeichnen der Daten mit Labels:

labeling_set_id	label	first_name	last_name	Geburtstag
ABC123	A	John	Doe	01.04.1980
ABC123	B	Jane	Smith	03.04.1980
ABC123	A	Johnny	Doe	01.04.1980
ABC123	A	Jon	Doe	01.04.1980
DEF345	A	Richard	Jones	11.12.1992
DEF345	A	Rich	Jones	12.11.1992
DEF345	B	Sarah	Jones	11.12.1992
DEF345	C	Richie	Jones Jr.	06.05.2017
DEF345	B	Sarah	Jones-Walker	11.12.1992
GHI678	A	Robert	Miller	03.01.1999
GHI678	A	Bob	Miller	03.01.1999
XYZABC	A	Wilhelm	Robinson	05.02.2001
XYZABC	B	Andreas	Robinson	05.02.1971

- Im obigen Beispiel identifizieren wir John/Johnny/Jon Doe as being a match and we teach the system that these records do not match Jane Smith. Separately, we teach the system that Richard and Rich Jones are the same person, but that these records are not a match to Sarah Jones/Jones-Walker und Richie Jones Jr.
- Wie Sie sehen, ist der Umfang der Labels auf die `labeling_set_id` beschränkt. Labels überschreiten daher keine `labeling_set_id`-Grenzen. Beispiel: Label „A“ für `labeling_set_id` 1 hat keinen Bezug zu Label „A“ für `labeling_set_id` 2.

- Wenn ein Datensatz keine Übereinstimmungen innerhalb eines Labeling-Satzes enthält, weisen Sie ihm ein eindeutiges Label zu. Jane Smith entspricht beispielsweise keinem Datensatz im Beschriftungssatz ABC123, sodass es der einzige Datensatz in diesem Beschriftungssatz mit der Bezeichnung B ist.
- Der Beschriftungssatz "GHI678" zeigt, dass ein Beschriftungssatz nur aus zwei Datensätzen bestehen kann, denen dieselbe Bezeichnung zugewiesen wurde, um zu zeigen, dass sie übereinstimmen. Ähnlich zeigt „XYZABC“ zwei Datensätze mit unterschiedlichen Labels, um zu zeigen, dass sie nicht übereinstimmen.
- Beachten Sie, dass manchmal ein Labeling-Satz keine Übereinstimmungen enthalten kann (d. h. Sie geben jedem Datensatz im Labeling-Satz ein anderes Label), oder ein Labeling-Satz könnte „alle gleich“ sein (Sie haben allen das gleiche Label gegeben). Dies ist in Ordnung, solange Ihre Labeling-Sätze zusammen Beispiele für Datensätze enthalten, die nach Ihren Kriterien „identisch“ sind und nicht.

> **Important**
>
> Vergewissern Sie sich, dass die IAM-Rolle, an die Sie übergeben AWS Glue hat Zugriff auf den Amazon S3 S3-Bucket, der die Label-Datei enthält. Konventionell AWS Glue Richtlinien gewähren Zugriff auf Amazon S3 S3-Buckets oder -Ordner, deren Namen das Präfix aws-glue - haben. Wenn sich die Labeling-Dateien an einem anderen Speicherort befinden, fügen Sie der IAM-Rolle die Berechtigung für diesen Speicherort hinzu.

Die Optimierung des maschinellen Lernens verwandelt sich in AWS Glue

Sie können Ihre Transformationen für maschinelles Lernen anpassen AWS Glue um die Ergebnisse Ihrer Datenbereinigungsaufgaben zu verbessern, um Ihre Ziele zu erreichen. Zur Verbesserung der Transformation können Sie ihr beibringen, eine Menge von Kennzeichnungen zu generieren, Kennzeichnungen hinzufügen und diese Schritte dann so oft wiederholen, bis die gewünschten Ergebnisse erzielt werden. Sie können eine Optimierung auch durch das Ändern einiger Machine-Learning-Parameter herbeiführen.

Weitere Informationen zu Machine Learning-Transformationen finden Sie unter Datensatzabgleich mit AWS Lake Formation FindMatches.

Themen

- Machine Learning-Messungen

- Entscheidung zwischen Präzision und Sensitivität
- Entscheidung zwischen Genauigkeit und Kosten
- Schätzen der Qualität von Übereinstimmungen mithilfe von Match-Konfidenzwerten
- Schulen der Transformation zum Suchen von Übereinstimmungen

Machine Learning-Messungen

Um die Messungen zu verstehen, die zur Optimierung der Machine Learning-Transformation verwendet werden, sollten Sie mit der folgenden Terminologie vertraut sein:

Tatsächliches Positiv (TP)

: Eine Datenübereinstimmung, die die Transformation korrekterweise ermittelt hat, manchmal auch als Treffer bezeichnet.

Tatsächliches Negativ (TN)

: Eine Nicht-Übereinstimmung der Daten, die die Transformation korrekterweise abgelehnt hat.

Falsches Positiv (FP)

: Eine Nicht-Übereinstimmung der Daten, die die Transformation fälschlicherweise als Übereinstimmung klassifiziert hat, manchmal auch als Fehlalarm bezeichnet.

Falsches Negativ (FN)

: Eine Übereinstimmung in den Daten, die die Transformation nicht gefunden hat, manchmal auch als Miss bezeichnet.

Weitere Informationen zur Terminologie, die beim Machine Learning verwendet wird, finden Sie unterConfusion matrix in Wikipedia.

Zur Optimierung Ihrer Machine Learning-Transformationen können Sie den Wert der folgenden Messungen innerhalb der Advanced properties (Erweiterten Eigenschaften) der Transformation ändern.

- Precision (Genauigkeit) misst, wie gut die Transformation echte positive Ergebnisse unter der Gesamtzahl der Datensätze findet, die sie als positiv identifiziert (echte positive und falsche positive). Weitere Informationen finden Sie unter Precision and recall in Wikipedia.

- Recall (Sensitivität) misst, wie gut die Transformation die tatsächlichen Positive aus der Gesamtzahl der Datensätze in den Quelldaten ermittelt. Weitere Informationen finden Sie unter Precision and recall in Wikipedia.
- Accuracy (Genauigkeit) misst, wie gut die Transformation tatsächliche Positive und tatsächliche Negative findet. Eine Erhöhung der Genauigkeit erfordert mehr maschinelle Ressourcen und erhöht die Kosten. Sie resultiert aber auch in einer erhöhten Sensitivität. Weitere Informationen finden Sie unter Accuracy and precision in Wikipedia.
- Cost (Kosten) misst, wie viele Datenverarbeitungsressourcen (und damit Geld) zum Ausführen der Transformation verbraucht werden.

Entscheidung zwischen Präzision und Sensitivität

Jede `FindMatches`-Transformation enthält einen `precision-recall`-Parameter. Sie verwenden diesen Parameter, um einen der folgenden Schritte festzulegen:

- Wenn Sie sich Gedanken darüber machen, dass die Transformation fälschlicherweise zwei Datensätze als übereinstimmend bezeichnet, obwohl sie tatsächlich nicht übereinstimmen, dann sollten Sie die Präzision verstärken.
- Wenn Sie sich Gedanken darüber machen, dass die Transformation zwei Datensätze nicht als übereinstimmend erkennt, die tatsächlich übereinstimmen, sollten Sie die Sensitivität verstärken.

Sie können diesen Kompromiss eingehen auf AWS Glue Konsole oder mit dem AWS Glue API-Operationen für maschinelles Lernen.

Wann Sie die Präzision erhöhen sollten

Erhöhen Sie die Präzision, wenn Sie sich Gedanken über das Risiko machen, dass`FindMatches` übereinstimmenden Datensätze anzeigt, obwohl sie in Wirklichkeit nicht übereinstimmen. Um die Präzision zu erhöhen, wählen Sie einen höheren Wert für den Trade-off zwischen Präzision und Sensitivität. Wenn Sie einen höheren Wert einstellen, benötigt die`FindMatches`-Transformation weitere Nachweise für die Entscheidung, ob ein Datensatzpaar übereinstimmt. Die Transformation wird dahingehend optimiert, dass sie feststellt, dass Datensätze nicht übereinstimmen.

Nehmen Sie zum Beispiel an, dass Sie `FindMatches` zur Erkennung von doppelten Elementen in einem Video-Katalog verwenden und einen höheren Wert des Präzisions-Sensitivitäts-Wert für die Transformation eingeben. Wenn Ihre Transformation fälschlicherweise feststellt, dass Star Wars: Eine neue Hoffnung identisch ist mit Star Wars: Das Imperium schlägt zurück, kann es passieren,

dass ein Kunde Eine neue Hoffnung möchte und stattdessen Das Imperium schlägt zurück angezeigt bekommt. Dies wäre eine schlechte Kundenerfahrung.

Wenn die Transformation jedoch nicht erkennt, dass Star Wars: Eine neue Hoffnung und Star Wars: Episode IV – Eine neue Hoffnung identisch sind, ist der Kunde vielleicht zunächst verwirrt, erkennt dann jedoch, dass sie identisch sind. Es wäre zwar ein Fehler, aber weniger schlimm als im vorherigen Szenario.

Wann Sie die Sensitivität erhöhen sollten

Erhöhen Sie die Sensitivität, wenn Sie sich Gedanken über das Risiko machen, dass die `FindMatches`-Transformationsergebnisse möglicherweise einen Datensatz nicht erkennen, der tatsächlich übereinstimmt. Um die Sensitivität zu erhöhen, wählen Sie einen niedrigeren Wert für den Präzisions-Sensitivitäts-Trade-off. Bei einem niedrigeren Wert sind für die `FindMatches`-Transformation weniger Nachweise für die Entscheidung erforderlich, dass ein Paar von Datensätzen übereinstimmt. Die Transformation ist dahingehend optimiert, dass sie Datensätze als übereinstimmend ansieht.

Dies kann zum Beispiel eine Priorität für eine Sicherheitsorganisation sein. Angenommen, Sie gleichen eine Liste von Kunden mit bekannten Betrügern ab und es ist wichtig festzustellen, ob ein Kunde ein Betrüger ist. Sie verwenden `FindMatches`, um die Liste der Betrüger mit der Liste der Kunden abzugleichen. Jedes Mal, wenn `FindMatches` eine Übereinstimmung zwischen den beiden Listen feststellt, wird ein menschlicher Prüfer zugewiesen, um zu prüfen, ob es sich bei der betreffenden Person tatsächlich um einen Betrüger handelt. Für Ihre Organisation ist vielleicht eine erhöhte Sensitivität wichtiger als die Präzision. Mit anderen Worten: Sie ziehen eine manuelle Prüfung durch die Auditoren vor und weisen einige Fälle zurück, in denen der Kunde kein Betrüger ist, als das Risiko einzugehen, einen Kunden nicht zu identifizieren, der tatsächlich auf der Betrügerliste steht.

So verbessern Sie sowohl die Präzision als auch die Sensitivität

Die beste Möglichkeit zur Verbesserung der Präzision und Sensitivität besteht darin, mehr Daten zu kennzeichnen. Je mehr Daten Sie kennzeichnen, desto mehr nimmt die allgemeine Genauigkeit der `FindMatches`-Transformation zu und verbessert damit sowohl die Präzision als auch die Sensitivität. Selbst bei der genauesten Transformation gibt es immer einen grauen Bereich, in dem Sie ausprobieren müssen, ob besser die Präzision oder die Sensitivität erhöht oder ein Wert in der Mitte gewählt werden soll.

Entscheidung zwischen Genauigkeit und Kosten

Jede `FindMatches`-Transformation enthält einen `accuracy-cost`-Parameter. Sie können diesen Parameter verwenden, um einen der folgenden Punkte festzulegen:

- Wenn Sie mehr Wert darauf legen, dass die Transformation korrekt meldet, dass zwei Datensätze übereinstimmen, sollten Sie den Faktor Genauigkeit verstärken.
- Wenn Sie mehr Wert auf die Kosten oder die Geschwindigkeit der Ausführung der Transformation legen, sollten Sie den Faktor geringere Kosten verstärken.

Sie können diesen Kompromiss eingehen auf AWS Glue Konsole oder mit dem AWS Glue API-Operationen für maschinelles Lernen.

Wann Sie der Genauigkeit den Vorzug geben sollten

Verstärken Sie die Genauigkeit, wenn Sie sich Gedanken über das Risiko machen, dass die `find matches`-Ergebnisse keine Übereinstimmungen enthalten. Um die Genauigkeit zu verstärken, wählen Sie einen höheren Genauigkeits-Kosten-Trade-off. Bei einem höheren Wert benötigt die`FindMatches`-Transformation mehr Zeit, um eine gründlichere Suche nach korrekt übereinstimmenden Datensätzen durchzuführen. Beachten Sie, dass dieser Parameter nicht die Wahrscheinlichkeit reduziert, dass ein nicht übereinstimmendes Datensatzpaar als übereinstimmend erkannt wird. Die Transformation ist dahingehend optimiert, dass sie mehr Zeit beim Suchen der Übereinstimmungen aufwendet.

Wann Sie die Kosten verstärken sollten

Verstärken Sie den Faktor „Kosten“, wenn Sie sich über die Kosten für die Ausführung der `find matches`-Transformation Gedanken machen und weniger darüber, wie viele Übereinstimmungen gefunden werden. Um die Kosten zu verstärken, wählen Sie einen niedrigeren Genauigkeits-Kosten-Trade-off. Bei einem niedrigeren Wert werden für die Ausführung der `FindMatches`-Transformation weniger Ressourcen benötigt. Die Transformation ist dahingehend optimiert, dass sie weniger Übereinstimmungen findet. Wenn die Ergebnisse bei der Verstärkung der niedrigeren Kosten akzeptabel sind, wählen Sie diese Einstellung.

So verstärken Sie sowohl den Faktor Genauigkeit als auch geringere Kosten

Es benötigt mehr Rechnerzeit, um mehrere Datensatzpaare daraufhin zu prüfen, ob sie möglicherweise übereinstimmen. Wenn Sie die Kosten senken möchten, ohne die Qualität zu mindern, empfehlen wir folgende Schritte:

- Entfernen Sie Datensätze in Ihrer Datenquelle, bei denen Ihnen nicht wichtig ist, ob sie übereinstimmen oder nicht.
- Entfernen Sie Spalten aus Ihrer Datenquelle, bei denen Sie sicher sind, dass sie bei der Entscheidungsfindung, ob ein Datensatzpaar übereinstimmt, nicht wichtig sind. Eine gute Möglichkeit, zu entscheiden, welche Spalten dies betrifft, ist zu überlegen, welche Spalten Ihre eigene Entscheidungsfindung darüber, ob eine Datensatzmenge übereinstimmt, nicht beeinflussen würden.

Schätzen der Qualität von Übereinstimmungen mithilfe von Match-Konfidenzwerten

Übereinstimmungswerte bieten eine Schätzung der Qualität der gefundenen Treffer, FindMatches um zwischen übereinstimmenden Datensätzen zu unterscheiden, bei denen das Modell des maschinellen Lernens sehr sicher, unsicher oder unwahrscheinlich ist. Ein Match-Konfidenzwert liegt zwischen 0 und 1, wobei ein höherer Punktestand eine höhere Ähnlichkeit bedeutet. Durch die Untersuchung von Match-Konfidenzwerten können Sie zwischen Clustern von Übereinstimmungen unterscheiden, in denen das System sehr zuversichtlich ist (die Sie möglicherweise zusammenführen möchten), Clustern, bei denen das System unsicher ist (die Sie möglicherweise von einem Menschen überprüfen lassen wollen) und Clustern, die das System für unwahrscheinlich hält (die Sie möglicherweise ablehnen).

Möglicherweise möchten Sie Ihre Trainingsdaten in Situationen anpassen, in denen Sie einen hohen Match-Konfidenzwert sehen, aber feststellen, dass es keine Übereinstimmungen gibt, oder in denen Sie eine niedrige Punktzahl sehen, aber feststellen, dass es tatsächlich Übereinstimmungen gibt.

Konfidenzwerte sind besonders nützlich, wenn umfangreiche industrielle Datensätze vorliegen, bei denen es nicht möglich ist, jede FindMatches Entscheidung zu überprüfen.

Werte für die Spielsicherheit sind verfügbar in AWS Glue Version 2.0 oder höher.

Generieren von Match-Konfidenzwerten

Sie können Match-Konfidenzwerte generieren, indem Sie beim Aufrufen der `FindMatches`- oder `FindIncrementalMatches`-API den booleschen Wert von `computeMatchConfidenceScores` auf „True“ setzen.

AWS Glue fügt der Ausgabe `column match_confidence_score` eine neue hinzu.

Beispiele für Match-Scoring

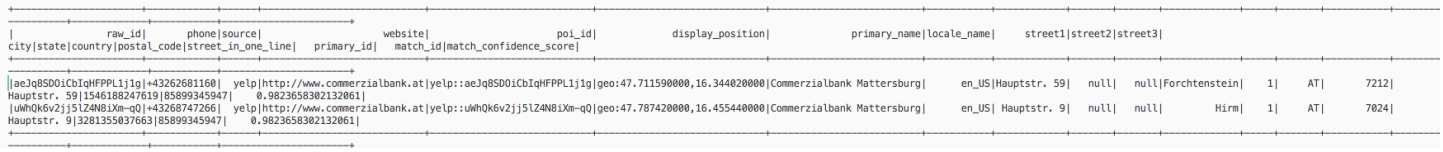
Betrachten Sie beispielsweise die folgenden übereinstimmenden Datensätze:

Ergebnis >= 0,9

Zusammenfassung der übereinstimmenden Datensätze:

```
  primary_id   |   match_id   | match_confidence_score

3281355037663    85899345947   0.9823658302132061
1546188247619    85899345947   0.9823658302132061
```

Details:



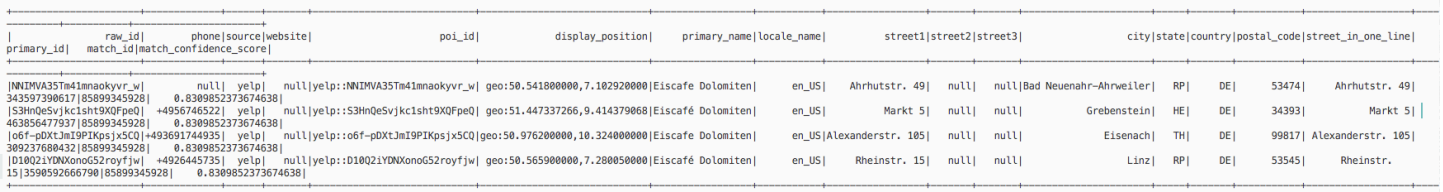
In diesem Beispiel sehen wir , dass zwei Datensätze sehr ähnlich sind und `display_position`, `primary_name` und `street name` gemeinsam haben.

Ergebnis >= 0,8 und Ergebnis < 0,9

Zusammenfassung der übereinstimmenden Datensätze:

```
  primary_id   |   match_id   | match_confidence_score

309237680432     85899345928   0.8309852373674638
3590592666790    85899345928   0.8309852373674638
343597390617     85899345928   0.8309852373674638
249108124906     85899345928   0.8309852373674638
463856477937     85899345928   0.8309852373674638
```

Details:



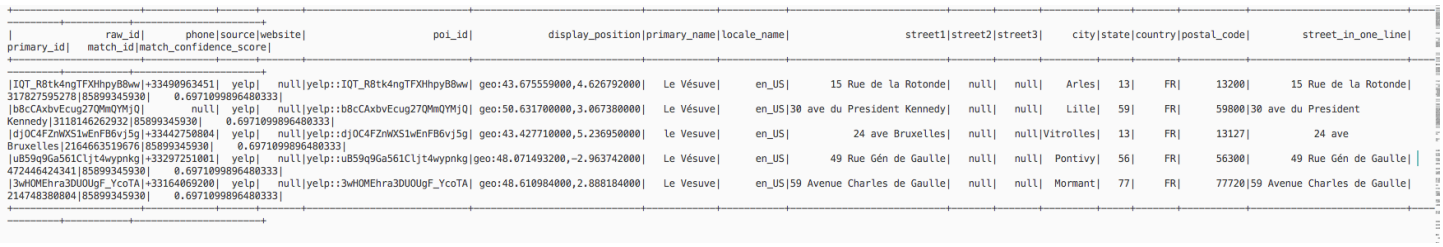
In diesem Beispiel sehen wir, dass diese Datensätze `primary_name` und `country` gemeinsam haben.

Ergebnis >= 0,6 und Ergebnis < 0,7

Zusammenfassung der übereinstimmenden Datensätze:

```
  primary_id  |   match_id   | match_confidence_score

2164663519676    85899345930   0.6971099896480333
 317827595278    85899345930   0.6971099896480333
 472446424341    85899345930   0.6971099896480333
3118146262932    85899345930   0.6971099896480333
 214748380804    85899345930   0.6971099896480333
```

Details:



In diesem Beispiel sehen wir, dass diese Datensätze nur `primary_name` gemeinsam haben.

Weitere Informationen finden Sie unter:

- Schritt 5: Hinzufügen und Ausführen eines Auftrags mit Ihrer Machine Learning-Transformation
- PySpark: FindMatches Klasse
- PySpark: FindIncrementalMatches Klasse
- Scala: FindMatches Klasse
- Scala: FindIncrementalMatches Klasse

Schulen der Transformation zum Suchen von Übereinstimmungen

Jede `FindMatches`-Transformation muss lernen, was als Übereinstimmung angesehen werden sollte und was nicht. Sie bringen Ihre Transformation bei, indem Sie einer Datei Beschriftungen hinzufügen und Ihre Auswahl hochladen AWS Glue.

Sie können diese Kennzeichnung auf dem orchestrieren AWS Glue Konsole oder mit dem AWS Glue API-Operationen für maschinelles Lernen.

Wie oft soll ich Kennzeichnungen hinzufügen? Wie viele Kennzeichnungen benötige ich?

Die Antworten auf diese Fragen sind meistens Ihre eigene Entscheidung. Sie müssen bewerten, ob `FindMatches` die Genauigkeitsstufe liefert, die Sie benötigen, und ob sich der zusätzliche Kennzeichnungsaufwand für Sie lohnt. Der beste Weg, dies zu entscheiden, besteht darin, sich die Kennzahlen „Präzision“, „Rückruf“ und „Fläche unter der Präzisionsrückrufkurve“ anzusehen, die Sie generieren können, wenn Sie Qualität schätzen auf der AWS Glue console. Nachdem Sie weitere Aufgabengruppen gekennzeichnet haben, führen Sie diese Metriken erneut aus und überprüfen Sie, ob sie sich verbessert haben. Wenn Sie nach der Kennzeichnung einiger Aufgabengruppen keine Verbesserung der Ihnen wichtigen Metriken feststellen können, hat die Transformationsqualität möglicherweise ein Plateau erreicht.

Warum werden sowohl die Kennzeichnung „tatsächliches Positiv“ und „tatsächliches Negativ“ benötigt?

Die `FindMatches`-Transformation benötigt sowohl positive als auch negative Beispiele, um zu lernen, was für Sie eine Übereinstimmung ist. Wenn Sie von `FindMatches` generierte Trainingsdaten kennzeichnen (z. B. mithilfe der OptionI do not have labels (Ich habe keine Kennzeichnungen), versucht `FindMatches`, eine Reihe von „Kennzeichnungsgruppen-IDs“ für Sie zu erstellen. Innerhalb jeder Aufgabe geben Sie einigen Datensätzen dieselbe „Kennzeichnung“ und anderen Datensätzen andere „Kennzeichnungen“. Mit anderen Worten, die Aufgaben sind im Allgemeinen nicht entweder alle gleich oder alle unterschiedlich (aber es ist in Ordnung, wenn eine bestimmte Aufgabe „gleich“ oder „nicht gleich“ ist).

Wenn Sie Ihre `FindMatches`-Transformation mithilfe der Option Upload Labels from S3 (Kennzeichnungen von S3 hochladen) schulen, versuchen Sie, sowohl Beispiele für übereinstimmende als auch für nicht übereinstimmende Datensätze einzubeziehen. Es ist akzeptabel, nur einen Typ zu haben. Diese Kennzeichnungen helfen Ihnen, eine genauere `FindMatches`-Transformation aufzubauen, aber Sie müssen trotzdem noch einige der Datensätze, die Sie generieren, mithilfe der Option Generate labeling file (Kennzeichnungsdatei generieren) kennzeichnen.

Wie kann ich erzwingen, dass die Transformation die Übereinstimmungen genau so ermittelt, wie es ihr beigebracht wurde?

Die `FindMatches`-Transformation lernt von den Kennzeichnungen, die Sie bereitstellen. Daher kann sie Datensatzpaare generieren, die die bereitgestellten Kennzeichnungen nicht respektieren. Um sicherzustellen, dass die `FindMatches` Transformation Ihre Beschriftungen berücksichtigt, wählen Sie EnforceProvidedLabelsin FindMatchesParameter.

Welche Techniken können Sie verwenden, wenn eine ML-Transformation Elemente als Übereinstimmungen identifiziert, die keine echten Übereinstimmungen sind?

Sie können die folgenden Techniken verwenden:

- Erhöhen Sie den`precisionRecallTradeoff` auf einen höheren Wert. Dies führt dazu, dass weniger Übereinstimmungen gefunden werden, aber es sollte auch Ihr großes Cluster unterteilen, wenn der erreichte Wert hoch genug ist.
- Nehmen Sie die Ausgabezeilen, die den falschen Ergebnissen entsprechen und formatieren sie diese als Kennzeichnungsgruppe (Entfernen der Spalte`match_id` und Hinzufügen der Spalten`labeling_set_id` und `label`). Falls erforderlich, nehmen Sie eine Unterteilung in mehrere Kennzeichnungsgruppen vor, um sicherzustellen, dass sich der Kennzeichner alle Kennzeichnungsgruppen merken kann, während er die Kennzeichnungen zuweist. Anschließend kennzeichnen Sie die übereinstimmenden Datensätze korrekt und laden die Kennzeichnungsdatei hoch und fügen sie den vorhandenen Kennzeichnungen hinzu. Dies kann Ihrer Transformation genügend Informationen geben, wonach sie suchen soll, um das Muster zu verstehen.
- (Erweitert) Schauen Sie sich schließlich die Daten an, um festzustellen, ob ein Muster zu erkennen ist, das das System nicht wahrnimmt. Verarbeiten Sie diese Daten mit dem Standard vor AWS Glue Funktionen zur Normalisierung der Daten. Markieren Sie das, was der Algorithmus lernen soll, indem Sie Daten trennen, von denen Sie wissen, dass sie in ihren eigenen Spalten eine andere Wichtigkeit haben. Oder erstellen Sie kombinierte Spalten aus Spalten, deren Daten einen Bezug zueinander haben.

Die Arbeit mit maschinellem Lernen transformiert

Sie können AWS Glue damit benutzerdefinierte Transformationen für maschinelles Lernen erstellen, mit denen Sie Ihre Daten bereinigen können. Diese Transformationen können Sie nutzen, wenn Sie einen Auftrag in der AWS Glue -Konsole anlegen.

Weitere Informationen zum Erstellen einer Machine Learning-Transformation finden Sie unter Datensatzabgleich mit AWS Lake Formation FindMatches.

Themen

- Transformieren von Eigenschaften
- Hinzufügen und Bearbeiten von Machine Learning-Transformationen
- Anzeigen von Transformationsdetails
- Transformationen mithilfe von Labels beibringen

Transformieren von Eigenschaften

Um eine bestehende Transformation für maschinelles Lernen einzusehen, melden Sie sich bei der AWS Management Console an und öffnen Sie die AWS Glue Konsole unter. https://console.aws.amazon.com/glue/ Wählen Sie in der -Konsole im Navigationsbereich unter Datenintegration und ETL die Optionen Datenklassifizierungstools > Datensatzabgleich aus.

Die Eigenschaften für jede Transformation:

Namen der Transformation

Der eindeutige Name, den Sie der Transformation bei der Erstellung gegeben haben.

ID

Eine eindeutige Kennung der Transformation.

Label count (Anzahl der Beschriftungen)

Die Anzahl der Beschriftungen in der Labeling-Datei, die als Unterstützung beim Erstellen der Transformation zur Verfügung gestellt wurde.

Status

Gibt an, ob die Transformation den Status Ready (Bereit) oder Needs training (Benötigt Training) hat. Um eine Machine-Learning-Transformation in einem Auftrag erfolgreich ausführen zu können, muss sie den Status Ready (Bereit) haben.

Erstellt

Das Datum, an dem die Transformation erstellt wurde.

Geändert

Das Datum, an dem die Transformation zuletzt aktualisiert wurde.

Beschreibung

Die Beschreibung, die für die Transformation bereitgestellt wurde, sofern vorhanden.

AWS Glue-Version

Die verwendete AWS Glue-Version.

ID ausführen

Der eindeutige Name, den Sie der Transformation bei der Erstellung gegeben haben.

Aufgabentyp

Der Typ der Machine Learning-Transformation, z. B. Find matching records (Übereinstimmende Datensätze suchen).

Status

Zeigt den Status der Aufgabenausführung an. Zu den möglichen Status gehören:

- Wird gestartet
- In Ausführung
- Wird angehalten
- Angehalten
- Erfolgreich
- Fehlgeschlagen
- Zeitüberschreitung

Fehler

Wenn der Status „Fehlgeschlagen“ lautet, wird eine Fehlermeldung mit einer Beschreibung der Ursache des Fehlers angezeigt.

Hinzufügen und Bearbeiten von Machine Learning-Transformationen

Sie können eine Transformation auf der AWS Glue-Konsole anzeigen, löschen, einrichten, trainieren oder optimieren. Wählen Sie das Kontrollkästchen neben der Transformation in der Liste aus, wählen Sie dann die OptionAktion und anschließend die Aktion, die Sie vornehmen möchten.

Erstellen einer neuen ML-Transformation

Um eine neue Machine-Learning-Transformation hinzuzufügen, wählen Sie Transformation erstellen. Folgen Sie den Anweisungen im Auftrag-hinzufügen-Assistenten. Weitere Informationen finden Sie unter Datensatzabgleich mit AWS Lake Formation FindMatches.

Schritt 1. Legen Sie Transformationseigenschaften fest.

1. Geben Sie den Namen und die Beschreibung ein (optional).
2. Optional können Sie die Sicherheitskonfiguration festlegen. Siehe Verwenden von Datenverschlüsselung mit Machine-Learning-Transformationen.
3. Legen Sie optional Einstellungen für die Aufgabenausführung fest. Mithilfe den Einstellungen für die Aufgabenausführung können Sie die Ausführung der Aufgabe individuell anpassen. Wählen

Sie den Worker-Typ, die Anzahl der Worker, das Aufgaben-Timeout (in Minuten), die Anzahl der Wiederholungsversuche und die AWS Glue-Version aus.

4. Legen Sie optional Tags fest. Tags sind Bezeichnungen, die Sie einer AWS Ressource zuweisen können. Jedes Tag besteht aus einem Schlüssel und einem optionalen Wert. Mithilfe von Tags können Sie Ihre Ressource durchsuchen und filtern oder Ihre AWS Kosten verfolgen.

Schritt 2. Wählen Sie Tabelle und Primärschlüssel aus.

1. Wählen Sie die AWS Glue-Katalogdatenbank und die Tabelle aus.
2. Wählen Sie einen Primärschlüssel aus der ausgewählten Tabelle. Die Primärschlüsselspalte enthält normalerweise eine eindeutige Kennung für jeden Datensatz in der Datenquelle.

Schritt 3. Wählen Sie Optimierungsoptionen aus.

1. Wählen Sie für Rückruf im Vergleich zu Präzision den Optimierungswert aus, um die Transformation so abzustimmen, dass Rückruf oder Präzision bevorzugt werden. Standardmäßig ist Ausgewogen ausgewählt. Sie können aber auch Rückruf oder Präzision bevorzugen oder Benutzerdefiniert auswählen und einen Wert zwischen 0,0 und 1,0 (einschließlich) eingeben.
2. Wählen Sie für Geringere Kosten im Vergleich zu Genauigkeit den Optimierungswert aus, um niedrigere Kosten oder Genauigkeit zu begünstigen, oder wählen Sie Benutzerdefiniert und geben Sie einen Wert zwischen 0,0 und 1,0 (einschließlich) ein.
3. Wählen Sie für Übereinstimmung erzwingen die Option Ausgabe zur Übereinstimmung mit Labels erzwingen, wenn Sie die ML-Transformation trainieren möchten, indem Sie die Ausgabe dazu erzwingen, mit den verwendeten Labels übereinzustimmen.

Schritt 4. Überprüfen und erstellen

1. Überprüfen Sie die Optionen für die Schritte 1 bis 3.
2. Wählen Sie Bearbeiten für jeden Schritt, der geändert werden muss. Wählen Sie Transformation erstellen, um den Assistenten zum Erstellen einer Transformation abzuschließen.

Verwenden von Datenverschlüsselung mit Machine-Learning-Transformationen

Wenn Sie eine Machine-Learning-Transformation zu AWS Glue hinzufügen, können Sie optional eine Sicherheitskonfiguration angeben, die der Datenquelle oder dem Datenziel zugeordnet ist. Wenn der

Amazon-S3-Bucket, der zum Speichern der Daten verwendet wird, mit einer Sicherheitskonfiguration verschlüsselt ist, geben Sie beim Erstellen der Transformation dieselbe Sicherheitskonfiguration an.

Sie können sich auch für die serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) entscheiden, um das Modell und die Beschriftungen zu verschlüsseln, um zu verhindern, dass Unbefugte es überprüfen können. Wenn Sie diese Option wählen, werden Sie aufgefordert, den AWS KMS key Namen auszuwählen, oder Sie können Enter a key ARN wählen. Wenn Sie den ARN für den KMS-Schlüssel eingeben, wird ein zweites Feld angezeigt, in dem Sie den KMS-Schlüssel-ARN eingeben können.

> **Note**
>
> Derzeit werden ML-Transformationen, die einen benutzerdefinierten Verschlüsselungsschlüssel verwenden, in den folgenden Regionen nicht unterstützt:
>
> - Asien-Pazifik (Osaka) – `ap-northeast-3`

Anzeigen von Transformationsdetails

Transformationseigenschaften anzeigen

Die Seite Transformationseigenschaften enthält Attribute Ihrer Transformation. Es zeigt Ihnen die Details zur Transformationsdefinition, einschließlich der folgenden:

- Transform name (Name der Transformation) zeigt den Namen der Transformation an.
- Type (Typ) führt die Art der Transformation auf.
- Status zeigt an, ob die Transformation bereit ist, in einem Skript verwendet zu werden.
- Force output to match labels (Ausgabe zwingen, Kennzeichnungen zuzuweisen) zeigt an, ob die Transformation die Ausgabe zwingt, die vom Benutzer bereitgestellten Kennzeichnungen zuzuweisen.
- Spark version (Spark-Version) bezieht sich auf die AWS Glue-Version, die Sie unter Task run properties (Eigenschaften für die Ausführung einer Aufgabe) beim Hinzufügen der Transformation ausgewählt haben. AWS Glue 1.0 und Spark 2.4 wird für die meisten Kunden empfohlen. Weitere Informationen finden Sie unter AWS Glue-Versionen.

Registerkarten „Verlauf“, „Qualität schätzen“ und „Tags“

Zu den Transformationsdetails gehören die Informationen, die Sie beim Erstellen der Transformation definiert haben. Um die Details einer Transformation anzuzeigen, wählen Sie die Transformation in der Liste Machine learning transforms (Machine Learning-Transformationen) aus und überprüfen Sie die Informationen auf den folgenden Registerkarten:

- Verlauf
- Schätzen der Qualität
- Tags

Verlauf

Die Registerkarte History (Verlauf) zeigt den Ausführungsverlauf Ihrer Aufgabe an. Mehrere Arten von Aufgaben werden ausgeführt, um eine Transformation zu schulen. Für jede Aufgabe enthalten die Ausführungsmetriken Folgendes:

- Run ID (Ausführungs-ID) ist ein von AWS Glue erstellter Bezeichner für jede Ausführung dieser Aufgabe.
- Task-Type (Aufgabentyp) zeigt die Art der ausgeführten Aufgabe.
- Status zeigt den Erfolg jeder aufgeführten Aufgabe an, wobei die neueste Aufgabe oben aufgeführt wird.
- Error (Fehler) zeigt die Details einer Fehlermeldung an, wenn die Ausführung nicht erfolgreich war.
- Start time (Startzeit) zeigt das Datum und die Uhrzeit (Ortszeit), an denen die Aufgabe gestartet wurde.
- Endzeit zeigt das Datum und die Uhrzeit (Ortszeit), an denen die Aufgabe beendet wurde.
- Logs (Protokolle) verlinkt sich mit den Protokollen, die für diese Auftragsausführung in `stdout` geschrieben wurden.

  Über den Link Logs gelangen Sie zu Amazon CloudWatch Logs. Dort können Sie die Details zu den Tabellen, die in der erstellt wurden, AWS Glue Data Catalog und zu allen aufgetretenen Fehlern einsehen. Sie können die Aufbewahrungsdauer Ihrer Protokolle auf der CloudWatch Konsole verwalten. Der Standardaufbewahrungszeitraum für Protokolle ist `Never Expire`. Weitere Informationen zum Ändern der Aufbewahrungsdauer finden Sie unter Ändern der Aufbewahrung von Protokolldaten in CloudWatch Protokollen im Amazon CloudWatch Logs-Benutzerhandbuch.

- Die Label-Datei zeigt einen Link zu Amazon S3 für eine generierte Beschriftungsdatei.

Schätzen der Qualität

Die Registerkarte Estimate Quality (Qualität schätzen) zeigt die Metriken, die Sie verwenden, um die Qualität der Transformation zu messen. Die Schätzungen werden berechnet, indem die Prognosen für die Transformationsübereinstimmung unter Verwendung einer Teilmenge Ihrer gekennzeichneten Daten mit den von Ihnen angegebenen Kennzeichnungen verglichen werden. Diese Schätzungen sind ungefähre Angaben. Sie können die Ausführung der Aufgabe Estimate quality (Qualität schätzen) aus dieser Registerkarte aufrufen.

Die Registerkarte Estimate quality (Qualität schätzen) zeigt die Metriken der letzten Ausführung von Estimate quality (Qualität schätzen) mit den folgenden Eigenschaften:

- Area under the Precision-Recall curve (Bereich unter der Präzisions-Sensitivitäts-Kurve) ist eine einzelne Zahl, die die obere Grenze der Gesamtqualität der Transformation schätzt. Sie ist unabhängig von der Wahl für den Präzisions-Sensitivitäts-Parameter. Höhere Werte weisen darauf hin, dass Sie einen attraktiveren Präzisions-Sensitivitäts-Trade-off haben.
- Precision (Präzision) schätzt, wie oft die Transformation korrekt ist, wenn sie eine Übereinstimmung prognostiziert.
- Recall upper limit (Obergrenze der Sensitivität) schätzt für eine tatsächliche Übereinstimmung, wie oft die Transformation die Übereinstimmung vorhersagt.
- F1 gibt eine Schätzung der Genauigkeit der Transformation zwischen 0 und 1 an, wobei 1 die beste Genauigkeit ist. Weitere Informationen finden Sie unter F1 score in Wikipedia.
- In der Tabelle Column importance (Bedeutung der Spalte) werden die Spaltennamen und die Bewertung für die Bedeutung jeder Spalte angezeigt. Anhand der Spaltenbedeutung können Sie verstehen, wie Spalten zu Ihrem Modell beitragen, indem Sie ermitteln, welche Spalten in Ihren Datensätzen am häufigsten für den Abgleich verwendet werden. Diese Daten können Sie dazu veranlassen, Ihr Labelset hinzuzufügen oder zu ändern, um die Bedeutung von Spalten zu erhöhen oder zu verringern.

  Die Spalte „Importance (Bedeutung)“ enthält eine numerische Bewertung für jede Spalte, da eine Dezimalzahl nicht größer als 1,0 ist.

Weitere Informationen zum Verständnis der Qualitätsschätzungen im Vergleich zur tatsächlichen Qualität finden Sie unter Qualitätsschätzungen versus end-to-end (wahre) Qualität.

Weitere Informationen zum Optimieren der Transformation finden Sie unter Die Optimierung des maschinellen Lernens verwandelt sich in AWS Glue.

Qualitätsschätzungen versus end-to-end (wahre) Qualität

AWS Glue schätzt die Qualität Ihrer Transformation, indem dem Machine-Learning-Modell eine Reihe von Datensatzpaaren präsentiert werden, für die Sie übereinstimmende Labels angegeben haben, die das Modell bisher jedoch nicht kannte. Diese Qualitätsschätzungen sind eine Qualitätsfunktion des Machine-Learning-Modells (abhängig von der Anzahl der Datensätze, die Sie für das Trainieren der Transformation gekennzeichnet haben). Der end-to-end oder wahre Erinnerungswert (der nicht automatisch durch den berechnet wird`ML transform`) wird auch durch den `ML transform` Filtermechanismus beeinflusst, der eine Vielzahl möglicher Übereinstimmungen mit dem maschinell erlernten Modell vorschlägt.

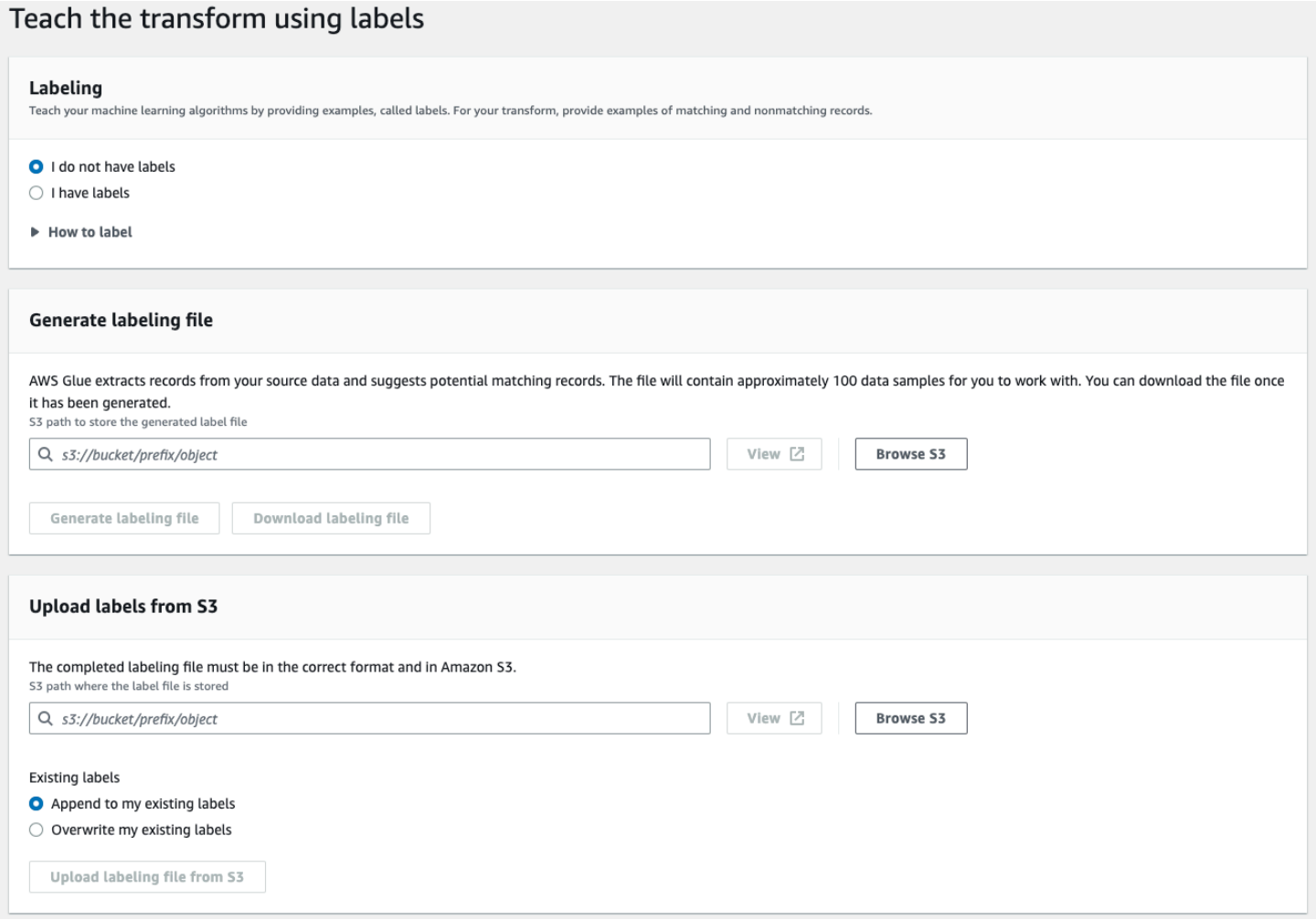
Sie können diese Filtermethode hauptsächlich durch Angabe des Optimierungswerts Niedrigere Kostengenauigkeit optimieren. Wenn sich der Optimierungswert der Genauigkeit annähert, führt das System eine gründlichere und aufwändigere Suche nach möglicherweise übereinstimmenden Datensatzpaaren durch. Es werden mehr Datensatzpaare in Ihr maschinell gelerntes Modell `ML transform` eingespeist, und Ihr end-to-end oder Ihr wahrer Erinnerungswert nähert sich der geschätzten Erinnerungsmetrik. Das hat zur Folge, dass Änderungen in der end-to-end Qualität Ihrer Matches aufgrund von Änderungen beim Kompromiss zwischen Kosten und Genauigkeit Ihrer Treffer in der Regel nicht in der Qualitätsschätzung berücksichtigt werden.

Tags

Tags sind Bezeichnungen, die Sie einer Ressource zuweisen können. AWS Jedes Tag besteht aus einem Schlüssel und einem optionalen Wert. Mithilfe von Tags können Sie Ihre Ressource durchsuchen und filtern oder Ihre AWS Kosten verfolgen.

Transformationen mithilfe von Labels beibringen

Sie können Ihrer ML-Transformation mithilfe von Beschriftungen (Beispiele) beibringen, indem Sie auf der Detailseite der ML-Transformation die Option Transformation beibringen auswählen. Wenn Sie Ihrem Machine-Learning-Algorithmus Beispiele (sogenannte Labels) beibringen, können Sie vorhandene Labels zur Verwendung auswählen oder eine Labeldatei erstellen.



- Beschriftung – Wenn Sie über Labels verfügen, wählen Sie Ich habe Labels. Wenn Sie keine Labels haben, können Sie trotzdem mit dem nächsten Schritt fortfahren, um eine Label-Datei zu generieren.
- Label-Datei generieren – AWS Glue extrahiert Datensätze aus Ihren Quelldaten und schlägt potenziell passende Datensätze vor. Sie wählen den Amazon-S3-Bucket aus, in dem die generierte Label-Datei gespeichert werden soll. Wählen Sie Label-Datei generieren, um den Vorgang zu starten. Wenn Sie fertig sind, wählen Sie Label-Datei herunterladen. Die heruntergeladene Datei enthält eine Spalte für Labels, in die Sie die Labels eintragen können.
- Labels aus Amazon S3 hochladen – Wählen Sie die fertige Label-Datei aus dem Amazon-S3-Bucket aus, in dem die Label-Datei gespeichert ist. Wählen Sie dann aus, ob Sie die Labels an Ihre vorhandenen Labels anfügen oder Ihre vorhandenen Labels überschreiben möchten. Wählen Sie Label-Datei aus Amazon S3 hochladen aus.

Tutorial: Erstellen einer Machine Learning-Transformation mit AWS Glue

Dieses Tutorial führt Sie durch die Aktionen zum Erstellen und Verwalten einer Machine Learning(ML)-Transformation mit AWS Glue. Bevor Sie dieses Tutorial verwenden, sollten Sie mit der Verwendung von vertraut sein AWS Glue Konsole zum Hinzufügen von Crawlern und Jobs und zum Bearbeiten von Skripten. Sie sollten auch wissen, wie Dateien gesucht und zur Amazon Simple Storage Service (Amazon S3)-Konsole heruntergeladen werden.

In diesem Beispiel erstellen Sie eine `FindMatches` Transformation, um übereinstimmende Datensätze zu finden, bringen ihr bei, wie übereinstimmende und nicht übereinstimmende Datensätze identifiziert werden, und verwenden sie in einem AWS Glue Arbeit. Das Tool AWS Glue job schreibt eine neue Amazon S3 S3-Datei mit einer zusätzlichen Spalte namens`match_id`.

Die in diesem Tutorial verwendeten Quelldaten befinden sich in einer Datei mit dem Namen `dblp_acm_records.csv`. Diese Datei ist eine modifizierte Version akademischer Publikationen (DBLP und ACM), die aus dem DBLP ACM-Originaldatensatz verfügbar sind. Die Datei `dblp_acm_records.csv` ist eine CSV-Datei (durch Kommas voneinander getrennte Werte) im UTF-8-Format ohne BOM (Byte-Order Mark, Markierung der Byte-Reihenfolge).

Eine zweite Datei, `dblp_acm_labels.csv`, ist eine Labeling-Beispieldatei mit übereinstimmenden und nicht übereinstimmenden Datensätzen, mit der die Transformation im Rahmen des Tutorials trainiert wird.

Themen

- Schritt 1: Crawlen der Quelldaten
- Schritt 2: Hinzufügen einer Machine Learning-Transformation
- Schritt 3: Trainieren Ihrer Machine Learning-Transformation
- Schritt 4: Beurteilen der Qualität Ihrer Machine Learning-Transformation
- Schritt 5: Hinzufügen und Ausführen eines Auftrags mit Ihrer Machine Learning-Transformation
- Schritt 6: Überprüfen der Ausgabedaten von Amazon S3

Schritt 1: Crawlen der Quelldaten

Durchforsten Sie zunächst die Amazon-S3-CSV-Quelldatei, um eine entsprechende Metadatendatei in Data Catalog zu erstellen.

> **Important**
>
> Um den Crawler dazu anzuweisen, eine Tabelle für ausschließlich die CSV-Datei zu erstellen, speichern Sie die CSV-Quelldaten in einem anderen Amazon-S3-Ordner als andere Dateien.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Crawlers (Crawler) und anschließend Add crawler (Crawler hinzufügen) aus.
3. Befolgen Sie die Anweisungen des Assistenten zum Erstellen und Ausführen eines Crawlers mit dem Namen `demo-crawl-dblp-acm` mit Ausgabe zur Datenbank `demo-db-dblp-acm`. Wenn der Assistent ausgeführt wird, erstellen Sie die Datenbank `demo-db-dblp-acm`, sofern sie noch nicht vorhanden ist. Wählen Sie einen Amazon S3 S3-Include-Pfad für Beispieldaten in der aktuellen AWS Region. Für `us-east-1` lautet der Amazon-S3-Include-Pfad zur Quelldatei beispielsweise `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/records/dblp_acm_records.csv`.

   Wenn der Crawler erfolgreich ist, erstellt er die Tabelle `dblp_acm_records_csv` mit den folgenden Spalten: id (ID), title (Titel), authors (Autoren), venue (Ort), year (Jahr) und source (Quelle).

Schritt 2: Hinzufügen einer Machine Learning-Transformation

Fügen Sie als Nächstes eine Machine Learning-Transformation hinzu, die auf dem Schema Ihrer Datenquellentabelle basiert, die von dem Crawler mit dem Namen `demo-crawl-dblp-acm` erstellt wurde.

1. Auf der AWS Glue Wählen Sie in der Konsole im Navigationsbereich unter Datenintegration und ETL die Optionen Datenklassifizierungstools > Datensatzabgleich und dann Transformation hinzufügen aus. Folgen Sie dem Assistenten beim Erstellen einer `Find matches`-Transformation mit den folgenden Eigenschaften.

   a. Geben Sie für Transform Name (Transformationsname) den Namen **`demo-xform-dblp-acm`** ein. Dies ist der Name der Transformation, der bei der Suche nach Übereinstimmungen in den Quelldaten verwendet wird.

b. Wählen Sie für die IAM-Rolle eine IAM-Rolle aus, die Zugriff auf die Amazon S3 S3-Quelldaten, die Label-Datei und AWS Glue API-Operationen. Weitere Informationen finden Sie unter Erstellen einer IAM-Rolle für AWS Glue im Entwicklerhandbuch für AWS Glue .

c. Wählen Sie als Datenquelle die Tabelle mit dem Namen dblp_acm_records_csv in der Datenbank aus. demo-db-dblp-acm

d. Wählen Sie für Primary key (Primärschlüssel) die Primärschlüsselspalte für die Tabelle id aus.

2. Wählen Sie im Assistenten Finish (Fertig stellen) und kehren Sie zur Liste ML transforms (ML-Transformationen) zurück.

Schritt 3: Trainieren Ihrer Machine Learning-Transformation

Als Nächstes trainieren Sie Ihre Machine Learning-Transformation bei der Verwendung der Labeling-Beispieldatei des Tutorials.

Eine Machine Language-Transformation kann erst dann in einem ETL-Auftrag (Extract, Transform and Load, Extrahieren, Transformieren und Laden) verwendet werden, wenn ihr Status Ready for use (Einsatzbereit) lautet. Sie müssen Ihre Transformation als Vorbereitung anhand von Beispielen übereinstimmender und nicht übereinstimmender Datensätze darin trainieren, wie übereinstimmende und nicht übereinstimmende Datensätze zu identifizieren sind. Zum Trainieren Ihrer Transformation können Sie Generate a label file (Eine Labling-Datei generieren) wählen, Labels hinzufügen und danach Upload label file (Labeling-Datei hochladen) wählen. In diesem Tutorial verwenden Sie die Beispiel-Labeling-Datei `dblp_acm_labels.csv`. Weitere Informationen über den Labeling-Prozess finden Sie unter Labeling.

1. Auf der AWS Glue Wählen Sie im Navigationsbereich der Konsole die Option Datensatzabgleich aus.

2. Wählen Sie die `demo-xform-dblp-acm`-Transformation aus und klicken Sie dann auf Action (Aktion) und Teach (Trainieren). Befolgen Sie die Anweisungen des Assistenten zum Trainieren Ihrer `Find matches`-Transformation.

3. Wählen Sie auf der Eigenschaftsseite der Transformation I have labels (Im Besitz von Labels). Wählen Sie einen Amazon S3 S3-Pfad zur Musteretikettierungsdatei in der aktuellen AWS Region. Beispiel: Für `us-east-1` laden Sie die bereitgestellte Labeling-Datei aus dem Amazon-S3-Pfad `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/labels/dblp_acm_labels.csv` mit der Option overwrite (Überschreiben), um vorhandene Labels zu

überschreiben. Die Kennzeichnungsdatei muss sich in Amazon S3 in derselben Region befinden wie die AWS Glue console.

Wenn Sie eine Label-Datei hochladen, wird eine Aufgabe gestartet in AWS Glue um die Beschriftungen hinzuzufügen oder zu überschreiben, die verwendet werden, um der Transformation beizubringen, wie die Datenquelle zu verarbeiten ist.

4. Wählen Sie auf der letzten Seite des Assistenten Finish (Fertig stellen) und kehren Sie zur Liste ML transforms (ML-Transformationen) zurück.

Schritt 4: Beurteilen der Qualität Ihrer Machine Learning-Transformation

Anschließend können Sie die Qualität Ihrer Machine Learning-Transformation beurteilen. Die Qualität hängt vom Ausmaß des bisher von Ihnen vorgenommenen Labeling ab. Weitere Informationen zur Beurteilung der Qualität finden Sie unter Schätzen der Qualität.

1. Auf dem AWS Glue Wählen Sie in der Konsole im Navigationsbereich unter Datenintegration und ETL die Option Datenklassifizierungstools > Datensatzabgleich aus.
2. Wählen Sie die Transformation `demo-xform-dblp-acm` und danach die Registerkarte Estimate quality (Qualität beurteilen) aus. Diese Registerkarte zeigt die aktuellen Qualitätsbeurteilungen für die Transformation an, sofern verfügbar.
3. Wählen Sie Estimate quality (Qualität beurteilen), um eine Aufgabe zur Beurteilung der Qualität der Transformation zu starten. Die Genauigkeit der Qualitätsbeurteilung basiert auf dem Labeling der Quelldaten.
4. Navigieren Sie zur Registerkarte History (Verlauf). In diesem Bereich werden Aufgabenausführungen für die Transformation aufgelistet, einschließlich der Aufgabe Estimating quality (Qualitätsbeurteilung). Um weitere Einzelheiten zur Ausführung zur erhalten, wählen Sie Logs (Protokolle). Stellen Sie sicher, dass der Status der Ausführung Succeeded (Erfolgreich) lautet, wenn sie beendet ist.

Schritt 5: Hinzufügen und Ausführen eines Auftrags mit Ihrer Machine Learning-Transformation

In diesem Schritt verwenden Sie Ihre Transformation für maschinelles Lernen, um einen Job hinzuzufügen und auszuführen in AWS Glue. Wenn die Transformation einsatzbereit **`demo-xform-dblp-acm`** ist, können Sie sie in einem ETL-Job verwenden.

1. Auf dem AWS Glue Wählen Sie im Navigationsbereich der Konsole Jobs aus.

2. Wählen Sie Add job (Auftrag hinzufügen) und befolgen Sie die Schritte im Assistenten zum Erstellen eines ETL-Spark-Auftrags mit einem generierten Skript. Wählen Sie die folgenden Eigenschaftswerte für Ihre Transformation aus:

   a. Wählen Sie als Name den Beispieljob in diesem Tutorial aus demo-etl-dblp-acm.

   b. Wählen Sie für die IAM-Rolle eine IAM-Rolle mit Zugriff auf die Amazon S3 S3-Quelldaten, die Label-Datei und AWS Glue API-Operationen. Weitere Informationen finden Sie unter Erstellen einer IAM-Rolle für AWS Glue im Entwicklerhandbuch für AWS Glue .

   c. Wählen Sie für ETL language (ETL-Sprache) die Option Scala aus. Das ist die Programmiersprache im ETL-Skript.

   d. Wählen demo-etl-dblp-acmSie als Namen der Skriptdatei. Das ist der Dateiname der Scala-Skript (identisch mit dem Auftragsnamen).

   e. Wählen Sie unter Data source (Datenquelle) als Quelle dblp_acm_records_csv aus. Die von Ihnen ausgewählte Datenquelle muss mit dem Datenquellen-Schema der Machine-Learning-Transformation übereinstimmen.

   f. Wählen Sie unter Transform type (Transformationstyp) die Option Find matching records (Übereinstimmende Datensätze suchen) aus, um einen Auftrag mit einer Machine Learning-Transformation zu erstellen.

   g. Deaktivieren Sie Remove duplicate records (Doppelte Datensätze entfernen). Doppelte Datensätze sollen nicht entfernt werden, da den geschriebenen Ausgabedatensätzen ein zusätziches `match_id`-Feld hinzugefügt wird.

   h. Wählen Sie für Transformieren die Transformation für maschinelles Lernen aus demo-xform-dblp-acm, die vom Job verwendet wird.

   i. Wählen Sie für Create tables in your data target (Tabellen in Ihren Zieldaten erstellen) das Erstellen von Tabellen mit den folgenden Eigenschaften:

      - Data store type (Datenspeichertyp) — **`Amazon S3`**
      - Format — **`CSV`**
      - Compression type (Komprimierungstyp) — **`None`**
      - Zielpfad — Der Amazon S3 S3-Pfad, in den die Ausgabe des Jobs geschrieben wird (in der aktuellen AWS Konsolenregion)

3. Wählen Sie Save Job und edit script (Auftrag speichern und Skript bearbeiten), um die Skript-Editor-Seite anzuzeigen.

4. Bearbeiten Sie das Skript, um eine Anweisung hinzufügen, die verlasst, dass die Ausgabe zu Target path (Zielpfad) in eine einzelne Partitionsdatei geschrieben wird. Fügen Sie diese

Anweisung unmittelbar nach der Anweisung ein, mit der die `FindMatches`-Tansformation ausgeführt wird. Die Anweisung gleicht der folgenden.

```
val single_partition = findmatches1.repartition(1)
```

Sie müssen die Anweisung `.writeDynamicFrame(findmatches1)` so ändern, dass die Ausgabe als `.writeDynamicFrame(single_partion)` geschrieben wird.

5. Wählen Sie nach dem Bearbeiten des Skripts Save (Speichern). Das geänderte Skript sieht in etwa wie der folgende Code, aber angepasst an Ihre Umgebung, aus.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.ml.FindMatches
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    // @type: DataSource
    // @args: [database = "demo-db-dblp-acm", table_name = "dblp_acm_records_csv",
 transformation_ctx = "datasource0"]
    // @return: datasource0
    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "demo-db-dblp-acm",
 tableName = "dblp_acm_records_csv", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
    // @type: FindMatches
    // @args: [transformId = "tfm-123456789012", emitFusion = false,
 survivorComparisonField = "<primary_id>", transformation_ctx = "findmatches1"]
    // @return: findmatches1
    // @inputs: [frame = datasource0]
```

```
    val findmatches1 = FindMatches.apply(frame = datasource0, transformId
 = "tfm-123456789012", transformationContext = "findmatches1",
 computeMatchConfidenceScores = true)


    // Repartition the previous DynamicFrame into a single partition.
    val single_partition = findmatches1.repartition(1)


    // @type: DataSink
    // @args: [connection_type = "s3", connection_options = {"path": "s3://aws-
glue-ml-transforms-data/sal"}, format = "csv", transformation_ctx = "datasink2"]
    // @return: datasink2
    // @inputs: [frame = findmatches1]
    val datasink2 = glueContext.getSinkWithFormat(connectionType =
 "s3", options = JsonOptions("""{"path": "s3://aws-glue-ml-transforms-
data/sal"}"""), transformationContext = "datasink2", format =
 "csv").writeDynamicFrame(single_partition)
    Job.commit()
  }
}
```

6. Wählen Sie Run job (Auftrag ausführen), um die Auftragsausführung zu starten. Überprüfen Sie den Status des Auftrags in der Auftragsliste. Wenn der Auftrag abgeschlossen ist, wurde die Registerkarte ML transform (ML-Transformation), History (Verlauf) um eine neueRun ID (Ausführungs-ID)-Zeile vom Typ ETL job (ETL-Auftrag) erweitert.
7. Navigieren Sie zur Registerkarte Jobs (Aufträge), History (Verlauf). In diesem Bereich werden Auftragsausführungen aufgelistet. Um weitere Einzelheiten zur Ausführung zur erhalten, wählen Sie Logs (Protokolle). Stellen Sie sicher, dass der Status der Ausführung Succeeded (Erfolgreich) lautet, wenn sie beendet ist.

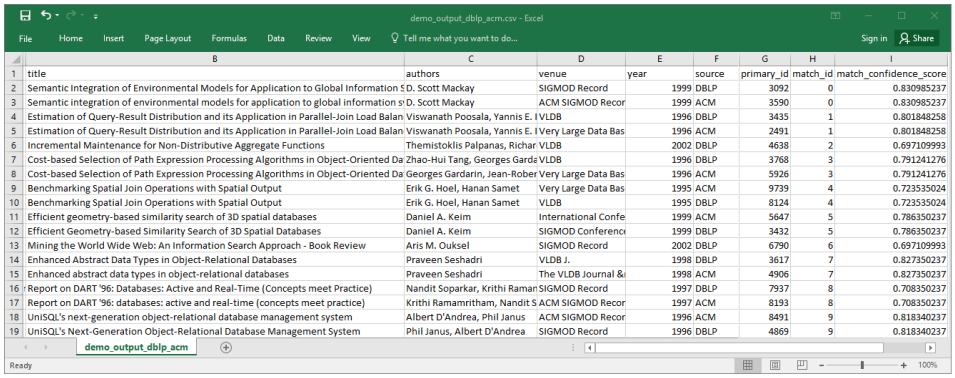
Schritt 6: Überprüfen der Ausgabedaten von Amazon S3

In diesem Schritt überprüfen Sie die Ausgabe der Auftragsausführung in dem Amazon-S3-Bucket, den Sie beim Hinzufügen des Auftrags ausgewählt haben. Sie können die Ausgabedatei auf Ihren lokalen Computer herunterladen und überprüfen, ob übereinstimmende Datensätze identifiziert wurden.

1. Öffnen Sie die Amazon S3 S3-Konsole unter https://console.aws.amazon.com/s3/.

2. Laden Sie die Ziel-Ausgabedatei des Auftrags `demo-etl-dblp-acm` herunter. Öffnen Sie die Datei in einer Tabellenkalkulationsanwendung (Sie müssen möglicherweise die Dateierweiterung `.csv` anhängen, damit die Datei ordnungsgemäß geöffnet wird).

   Die folgende Abbildung zeigt einen Ausschnitt der Ausgabe in Microsoft Excel.



	B	C	D	E	F	G	H	I
1	title	authors	venue	year	source	primary_id	match_id	match_confidence_score
2	Semantic Integration of Environmental Models for Application to Global Information S	D. Scott Mackay	SIGMOD Record	1999	DBLP	3092	0	0.830985237
3	Semantic integration of environmental models for application to global information s	D. Scott Mackay	ACM SIGMOD Recor	1999	ACM	3590	0	0.830985237
4	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	VLDB	1996	DBLP	3435	1	0.801848258
5	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	Very Large Data Bas	1996	ACM	2491	1	0.801848258
6	Incremental Maintenance for Non-Distributive Aggregate Functions	Themistoklis Palpanas, Richar	VLDB	2002	DBLP	4638	2	0.697109993
7	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Zhao-Hui Tang, Georges Garda	VLDB	1996	DBLP	3768	3	0.791241276
8	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Georges Gardarin, Jean-Rober	Very Large Data Bas	1996	ACM	5926	3	0.791241276
9	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	Very Large Data Bas	1995	ACM	9739	4	0.723535024
10	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	VLDB	1995	DBLP	8124	4	0.723535024
11	Efficient geometry-based similarity search of 3D spatial databases	Daniel A. Keim	International Confe	1999	ACM	5647	5	0.786350237
12	Efficient Geometry-based Similarity Search of 3D Spatial Databases	Daniel A. Keim	SIGMOD Conferenc	1999	DBLP	3432	5	0.786350237
13	Mining the World Wide Web: An Information Search Approach - Book Review	Aris M. Ouksel	SIGMOD Record	2002	DBLP	6790	6	0.697109993
14	Enhanced Abstract Data Types in Object-Relational Databases	Praveen Seshadri	VLDB J.	1998	DBLP	3617	7	0.827350237
15	Enhanced abstract data types in object-relational databases	Praveen Seshadri	The VLDB Journal &	1998	ACM	4906	7	0.827350237
16	Report on DART '96: Databases: Active and Real-Time (Concepts meet Practice)	Nandit Soparkar, Krithi Raman	SIGMOD Record	1997	DBLP	7937	8	0.708350237
17	Report on DART '96: databases: active and real-time (concepts meet practice)	Krithi Ramamritham, Nandit S	ACM SIGMOD Recor	1997	ACM	8193	8	0.708350237
18	UniSQL's next-generation object-relational database management system	Albert D'Andrea, Phil Janus	ACM SIGMOD Recor	1996	ACM	8491	9	0.818340237
19	UniSQL's Next-Generation Object-Relational Database Management System	Phil Janus, Albert D'Andrea	SIGMOD Record	1996	DBLP	4869	9	0.818340237

   Die Datenquelle und die Zieldatei besitzen beide 4 911 Datensätze. Die Transformation `Find matches` fügt jedoch eine weitere Spalte mit dem Namen `match_id` zur Identifizierung übereinstimmender Datensätze in der Ausgabe hinzu. Zeilen mit derselben `match_id` werden als übereinstimmende Datensätze angesehen. Der `match_confidence_score` ist eine Zahl zwischen 0 und 1, die eine Schätzung der Qualität der von `Find matches` gefundenen Übereinstimmungen liefert.

3. Sortieren Sie die Ausgabedatei nach `match_id`, damit übereinstimmende Datensätze leicht ersichtlich sind. Vergleichen Sie die Werte in den anderen Spalten, um festzustellen, ob Sie den Ergebnissen der Transformation `Find matches` zustimmen. Wenn dies nicht der Fall ist, können Sie die Transformation durch Hinzufügen weiterer Labels weiter trainieren.

   Sie können die Datei auch nach einem anderen Feld, z. B. `title`, sortieren, um festzustellen, ob Datensätze mit ähnlichen Titeln dieselbe `match_id` besitzen.

Inkrementelle Übereinstimmungen finden

Mithilfe dem Feature „FindMatches“ können Sie doppelte oder übereinstimmende Datensätze in Ihrem Dataset identifizieren, auch wenn die Datensätze nicht über eine gemeinsame eindeutige Kennung verfügen und keine Felder exakt übereinstimmen. Die erste Version der Suchübereinstimmungen transformiert übereinstimmende Datensätze innerhalb eines einzelnen Datensatzes. Wenn Sie dem Datensatz neue Daten hinzufügen, mussten Sie sie mit dem vorhandenen sauberen Datensatz zusammenführen und die Übereinstimmung mit dem vollständig zusammengeführten Datensatz erneut ausführen.

Mit dem Feature zur inkrementellen Übereinstimmung können inkrementelle Datensätze leichter mit vorhandenen übereinstimmenden Datensätzen abgeglichen werden. Angenommen, Sie möchten potenzielle Daten mit vorhandenen Kundendatensätzen abgleichen. Die Fähigkeit zur inkrementellen Übereinstimmung bietet Ihnen die Flexibilität, Hunderttausende neuer Interessenten mit einer bestehenden Datenbank von Interessenten und Kunden abzugleichen, indem Sie die Ergebnisse in einer einzigen Datenbank oder Tabelle zusammenführen. Durch den Abgleich nur zwischen den neuen und den vorhandenen Datensätzen reduziert die Optimierung der Suche nach inkrementellen Übereinstimmungen die Berechnungszeit, was auch die Kosten senkt.

Die Verwendung der inkrementellen Übereinstimmung ähnelt „Find Matches“, wie unter Tutorial: Erstellen einer Machine Learning-Transformation mit AWS Glue beschrieben. In diesem Thema werden nur die Unterschiede bei der inkrementellen Übereinstimmung identifiziert.

Weitere Informationen finden Sie im Blog-Beitrag zu Inkrementelle Datenübereinstimmung.

Ausführen eines Auftrags zu inkrementellen Übereinstimmungen

Nehmen wir Folgendes an:

- Sie haben den vorhandenen Datensatz in die Tabelle first_records gecrawlt. Der Datensatz first_records muss ein übereinstimmender Datensatz oder die Ausgabe des übereinstimmenden Auftrags sein.
- Sie haben eine Transformation für die Suche nach Übereinstimmungen mit erstellt und trainiert AWS Glue Version 2.0. Dies ist die einzige Version von AWS Glue das unterstützt inkrementelle Treffer.
- Die ETL-Sprache ist Scala. Python wird ebenfalls unterstützt.
- Das bereits generierte Modell heißt `demo-xform`.

1. Crawlen Sie den inkrementellen Datensatz in die Tabelle second_records.
2. Auf dem AWS Glue Wählen Sie im Navigationsbereich der Konsole Jobs aus.
3. Wählen Sie Add job (Auftrag hinzufügen) und befolgen Sie die Schritte im Assistenten zum Erstellen eines ETL-Spark-Auftrags mit einem generierten Skript. Wählen Sie die folgenden Eigenschaftswerte für Ihre Transformation aus:

    a. Für Name wählen Sie demo-etl aus.

    b. Wählen Sie für die IAM-Rolle eine IAM-Rolle mit Zugriff auf die Amazon S3 S3-Quelldaten, die Label-Datei und AWS Glue API-Operationen.

c. Wählen Sie für ETL language (ETL-Sprache) die Option Scala aus.

d. Wählen Sie unter Script file name (Skript-Dateiname) als Namen demo-etl aus. Dies ist der Dateiname des Scala-Skripts.

e. Für Datenquelle wählen Sie first_records aus. Die von Ihnen ausgewählte Datenquelle muss mit dem Datenquellen-Schema der Machine-Learning-Transformation übereinstimmen.

f. Wählen Sie unter Transform type (Transformationstyp) die Option Find matching records (Übereinstimmende Datensätze suchen) aus, um einen Auftrag mit einer Machine Learning-Transformation zu erstellen.

g. Wählen Sie die Option für den inkrementellen Abgleich und für Datenquelle die Tabelle mit dem Namen second_records aus.

h. Wählen Sie unter Transform (Transformation) die von diesem Auftrag verwendete Machine Learning-Transformation demo-xform aus.

i. Klicken Sie auf Erstellen von Tabellen in Ihrem Datenziel oder Verwenden von Tabellen im Data Catalog und Aktualisieren Ihres Datenziels.

4. Wählen Sie Save Job und edit script (Auftrag speichern und Skript bearbeiten), um die Skript-Editor-Seite anzuzeigen.

5. Wählen Sie Run job (Auftrag ausführen), um die Auftragsausführung zu starten.

Verwendung FindMatches in einem visuellen Job

Um die FindMatchesTransformation in zu verwenden AWS Glue Studio, können Sie den Custom Transform-Knoten verwenden, der die FindMatches API aufruft. Weitere Informationen zur Verwendung einer benutzerdefinierten Transformation finden Sie unter Erstellen einer benutzerdefinierten Transformation

> **Note**
>
> Derzeit funktioniert die FindMatches API nur mit`Glue 2.0`. Um einen Job mit der benutzerdefinierten Transformation auszuführen, die die FindMatches API aufruft, stellen Sie sicher, dass AWS Glue Die Version befindet `Glue 2.0` sich auf der Registerkarte Jobdetails. Wenn die Version von AWS Glue ist dies nicht der `Glue 2.0` Fall, schlägt der Job zur Laufzeit mit der folgenden Fehlermeldung fehl: „Der Name '' kann nicht aus FindMatches 'awsglueml.transforms' importiert werden“.

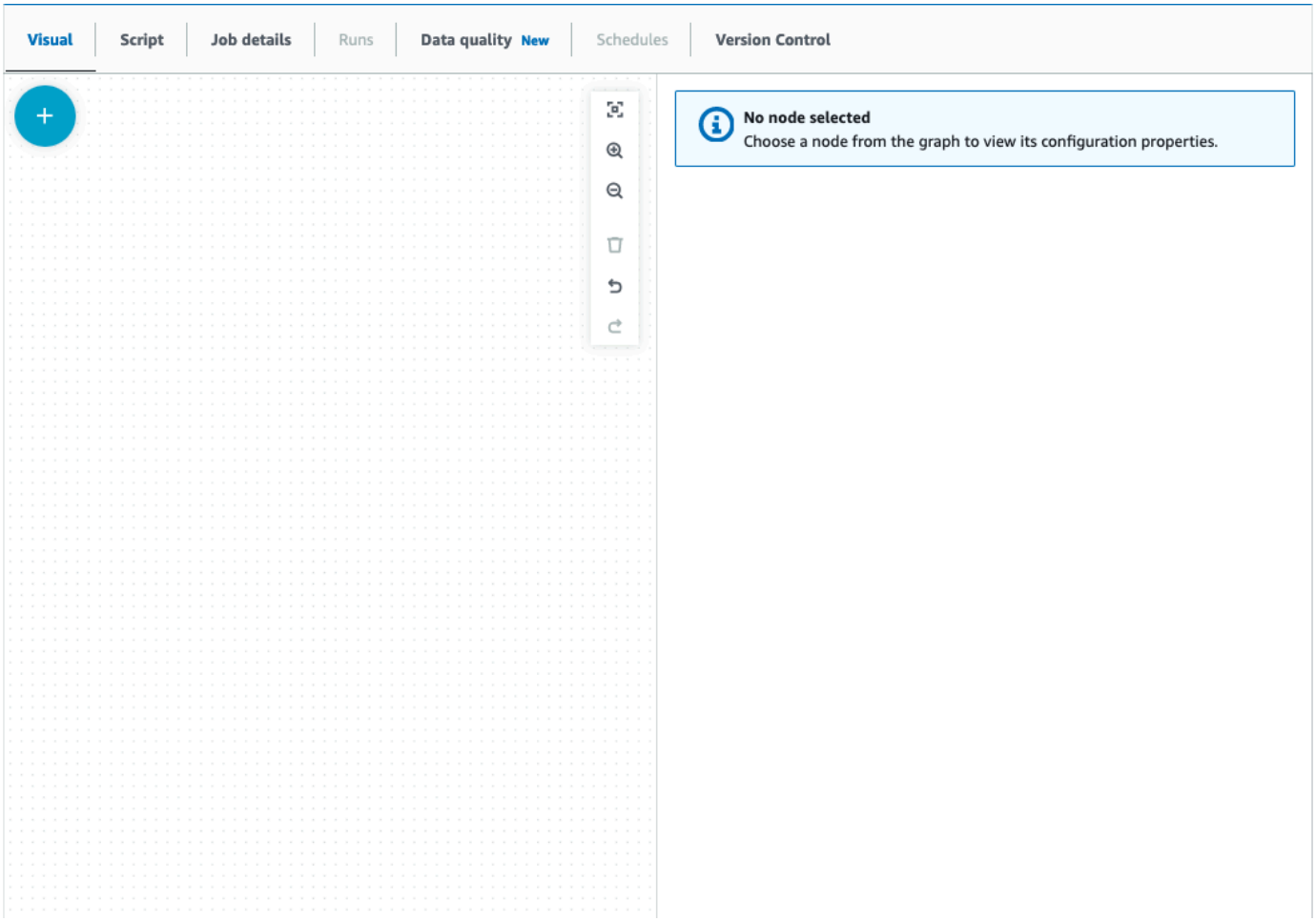
Voraussetzungen

- Um die Transformation „Treffer suchen“ zu verwenden, öffnen Sie die AWS Glue Studio Konsole unterhttps://console.aws.amazon.com/gluestudio/.
- Erstellen Sie eine Machine-Learning-Transformation. Beim Erstellen wird eine transformId generiert. Sie benötigen diese ID für die folgenden Schritte. Weitere Informationen darüber, wie Sie eine Machine Learning-Transformation erstellen, finden Sie unter Hinzufügen und Bearbeiten von Machine-Learning-Transformationen.

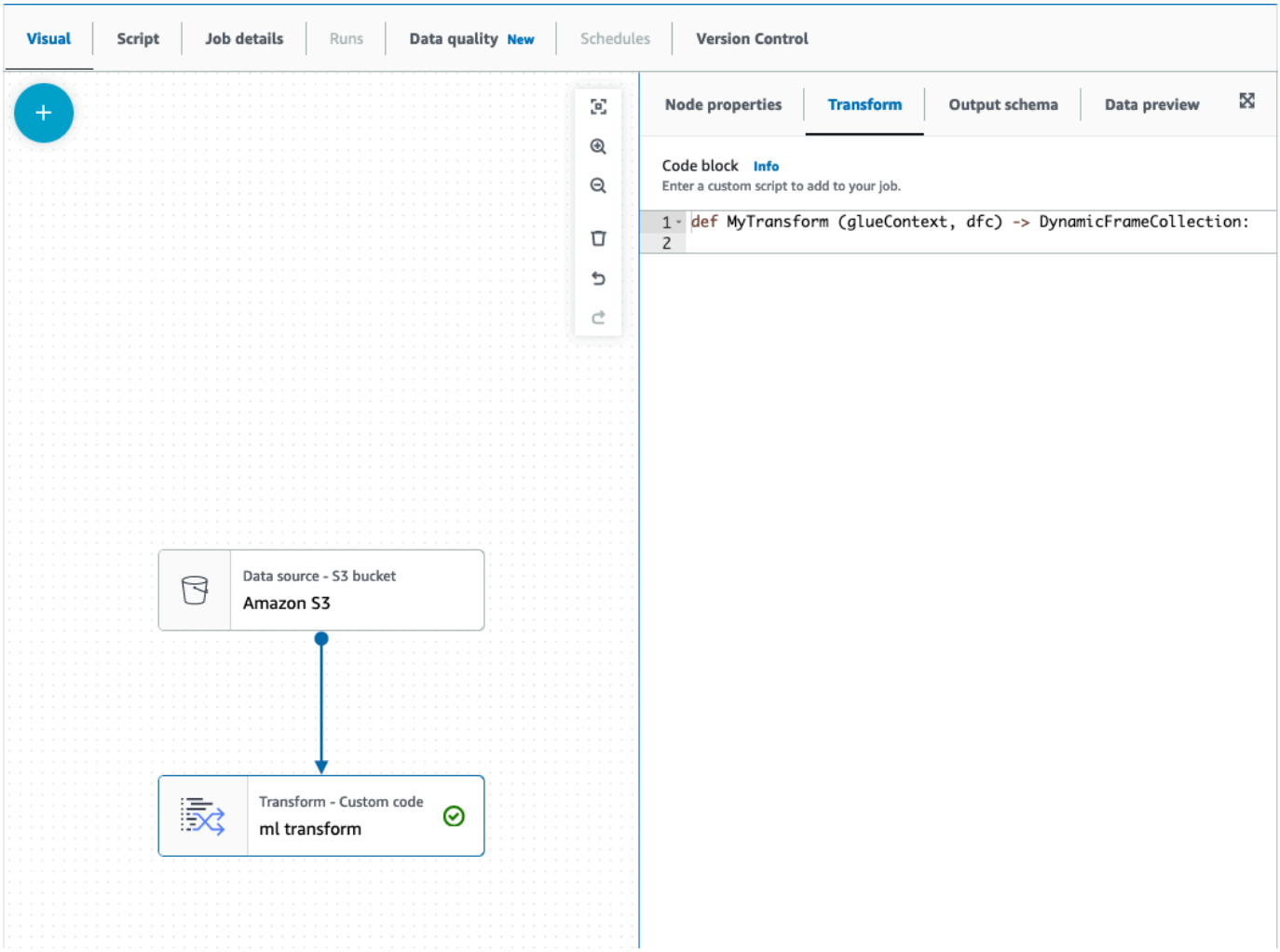
Eine FindMatches Transformation hinzufügen

Um eine FindMatches Transformation hinzuzufügen:

1. Im AWS Glue Studio Öffnen Sie im Job-Editor das Ressourcen-Panel, indem Sie auf das Kreuzsymbol in der oberen linken Ecke des visuellen Job-Diagramms klicken und eine Datenquelle auswählen, indem Sie auf die Registerkarte Daten klicken. Dies ist die Datenquelle, die Sie auf Übereinstimmungen überprüfen möchten.



2. Wählen Sie den Knoten Datenquelle, öffnen Sie das Bedienfeld Ressourcen, indem Sie auf das Kreuzsymbol in der oberen linken Ecke des visuellen Auftragsdiagramms klicken, und suchen Sie nach „benutzerdefinierte Transformation“. Wählen Sie den Knoten Benutzerdefinierte Transformation, um ihn dem Diagramm hinzuzufügen. Die Benutzerdefinierte Transformation ist mit dem Datenquellenknoten verknüpft. Ist dies nicht der Fall, können Sie auf den Knoten Benutzerdefinierte Transformation klicken und die Registerkarte Knoteneigenschaften auswählen. Wählen Sie dann unter Übergeordnete Knoten die Datenquelle aus.
3. Klicken Sie im visuellen Diagramm auf den Knoten Benutzerdefinierte Transformation, wählen Sie dann die Registerkarte Knoteneigenschaften aus und benennen Sie die benutzerdefinierte Transformation. Es wird empfohlen, die Transformation umzubenennen, damit der Name der Transformation im visuellen Diagramm leicht erkennbar ist.
4. Wählen Sie die Registerkarte Transformation, auf der Sie den Code-Block bearbeiten können. Hier kann der Code zum Aufrufen der FindMatches API hinzugefügt werden.



Der Code-Block enthält vorab ausgefüllten Code, um Ihnen den Einstieg zu erleichtern. Überschreiben Sie den vorab ausgefüllten Code mit der folgenden Vorlage. Die Vorlage verfügt über einen Platzhalter für die transformId, den Sie bereitstellen können.

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dynf = dfc.select(list(dfc.keys())[0])
    from awsglueml.transforms import FindMatches
    findmatches = FindMatches.apply(frame = dynf, transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

5. Klicken Sie im visuellen Diagramm auf den Knoten Benutzerdefinierte Transformation und öffnen Sie das Bedienfeld Ressourcen, indem Sie auf das Kreuzsymbol in der oberen linken Ecke des visuellen Auftragsdiagramms klicken und nach „Aus Sammlung auswählen“ suchen. Die

Standardauswahl muss nicht geändert werden, da es DynamicFrame in der Sammlung nur eine gibt.

6. Sie können weitere Transformationen hinzufügen oder das Ergebnis speichern, das nun um die zusätzlichen Spalten zum Suchen nach Übereinstimmungen erweitert wird. Wenn Sie diese neuen Spalten in nachgelagerten Transformationen referenzieren möchten, müssen Sie sie dem Ausgabeschema der Transformation hinzufügen. Am einfachsten geht das, indem Sie die Registerkarte Datenvorschau auswählen und dann auf der Registerkarte Schema die Option „Schema zu Datenvorschau verwenden“ auswählen.
7. Zur Anpassung können Sie zusätzliche Parameter hinzufügen FindMatches, die an die Methode „Anwenden“ übergeben werden. Siehe FindMatches Klasse.

Eine FindMatches inkrementelle Transformation hinzufügen

Bei inkrementellen Übereinstimmungen entspricht der Vorgang dem Hinzufügen einer FindMatches Transformation mit den folgenden Unterschieden:

- Anstelle eines übergeordneten Knotens für die benutzerdefinierte Transformation benötigen Sie zwei übergeordnete Knoten.
- Der erste übergeordnete Knoten sollte der Datensatz sein.
- Der zweite übergeordnete Knoten sollte der inkrementelle Datensatz sein.

  Ersetzen Sie im Vorlagen-Codeblock die `transformId` durch Ihre `transformId`:

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dfs = list(dfc.values())
    dynf = dfs[0]
    inc_dynf = dfs[1]
    from awsglueml.transforms import FindIncrementalMatches
    findmatches = FindIncrementalMatches.apply(existingFrame = dynf, incrementalFrame
 = inc_dynf,
                                    transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

- Optionale Parameter finden Sie unter FindIncrementalMatches Klasse.

Migrieren Sie Apache Spark-Programme zu AWS Glue

Apache Spark ist eine Open-Source-Plattform für verteilte Rechenlasten, die für große Datenmengen ausgeführt werden. AWS Glue nutzt die Fähigkeiten von Spark, um ein optimiertes ETL-Erlebnis zu bieten. Sie können Spark-Programme zu migrieren AWS Glue , um unsere Funktionen zu nutzen. AWS Glue bietet dieselben Leistungsverbesserungen, die Sie von Apache Spark auf Amazon EMR erwarten würden.

Spark-Code ausführen

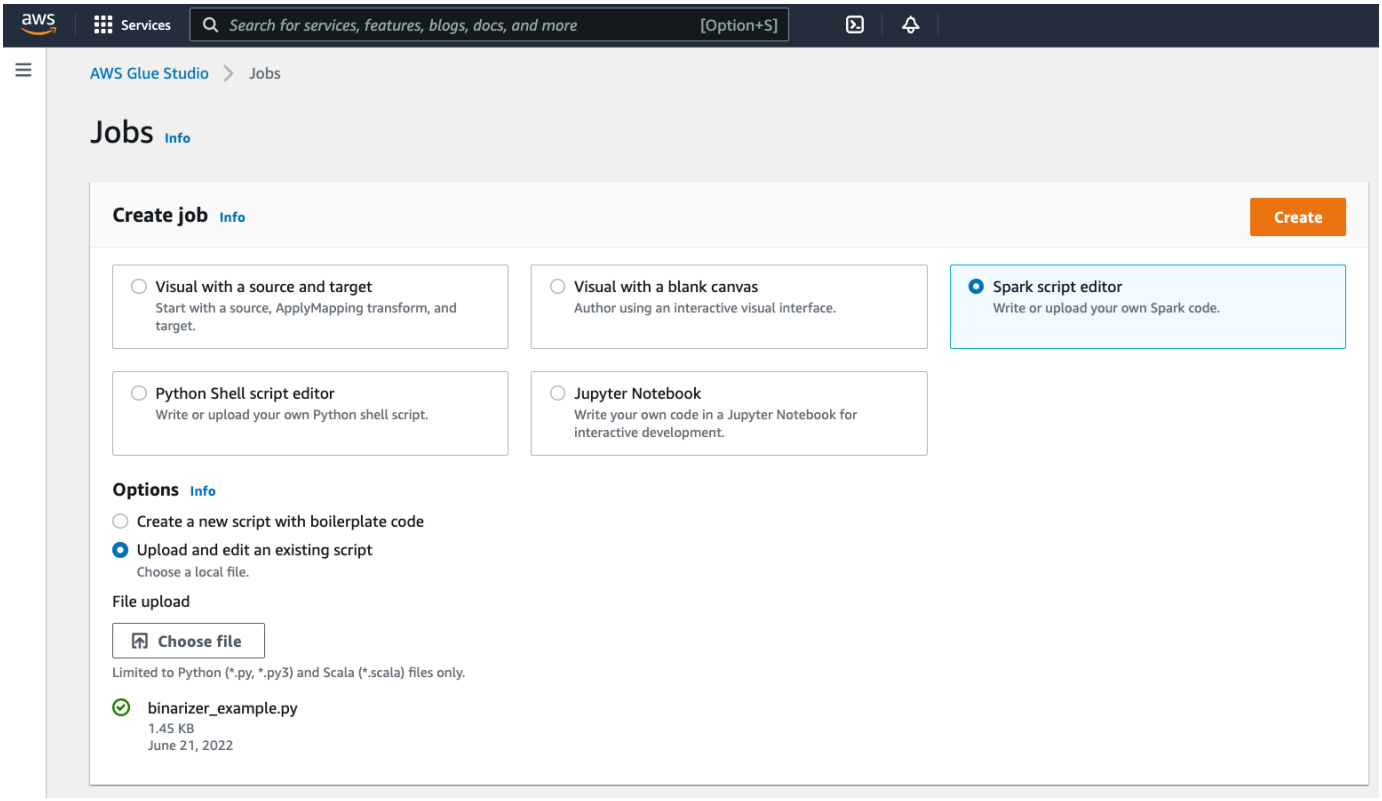
Nativer Spark-Code kann standardmäßig in einer AWS Glue Umgebung ausgeführt werden. Skripte werden oft entwickelt, indem ein Teil des Codes iterativ geändert wird, ein Workflow, der für eine interaktive Sitzung geeignet ist. Bestehender Code eignet sich jedoch besser für die Ausführung in einem AWS Glue Job, sodass Sie Protokolle und Metriken für jeden Skriptlauf planen und konsistent abrufen können. Sie können ein vorhandenes Skript über die Konsole hochladen und bearbeiten.

1. Eignen Sie sich die Quelle Ihres Skripts an. In diesem Beispiel verwenden Sie ein Beispielskript aus dem Apache Spark-Repository. Binarizer-Beispiel
2. Erweitern Sie in der AWS Glue Konsole den linken Navigationsbereich und wählen Sie ETL > Jobs

   Wählen Sie im Feld Erstellen von Aufträgen Spark-Script-Editor aus. Es wird ein Optionen-Abschnitt erscheinen. Wählen Sie unter Optionen Upload und Bearbeiten eines vorhandenes Skripts aus.

   Es wird ein Datei-Upload-Abschnitt erscheinen. Klicken Sie unter Datei-Upload auf Datei auswählen. Ihre Systemdateiauswahl wird erscheinen. Navigieren Sie zu dem Speicherort, an dem Sie `binarizer_example.py` gespeichert haben, wählen Sie es aus und bestätigen Sie Ihre Auswahl.

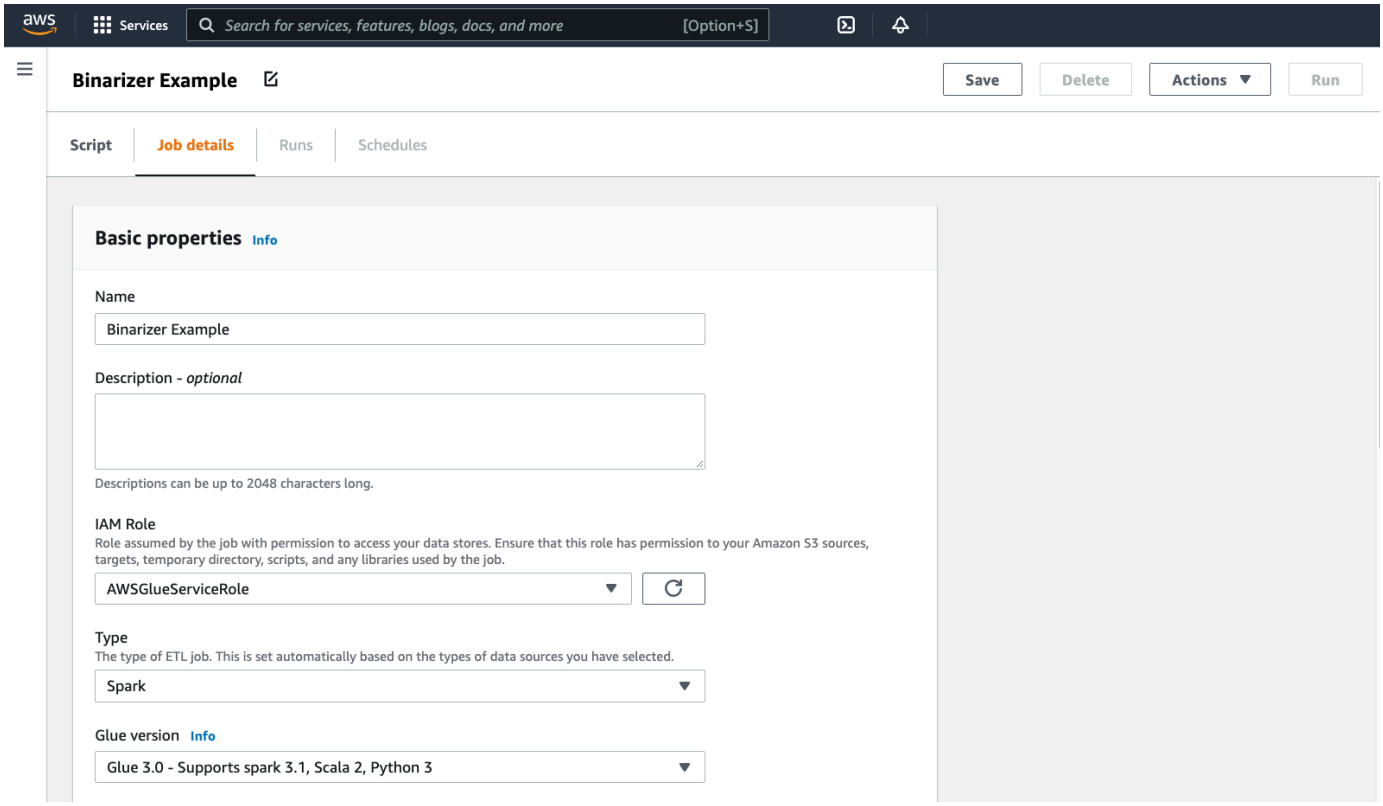
   Eine Schaltfläche für Erstellen wird in der Kopfzeile des Fensters Erstellen von Aufträgen erscheinen. Klicken Sie darauf.



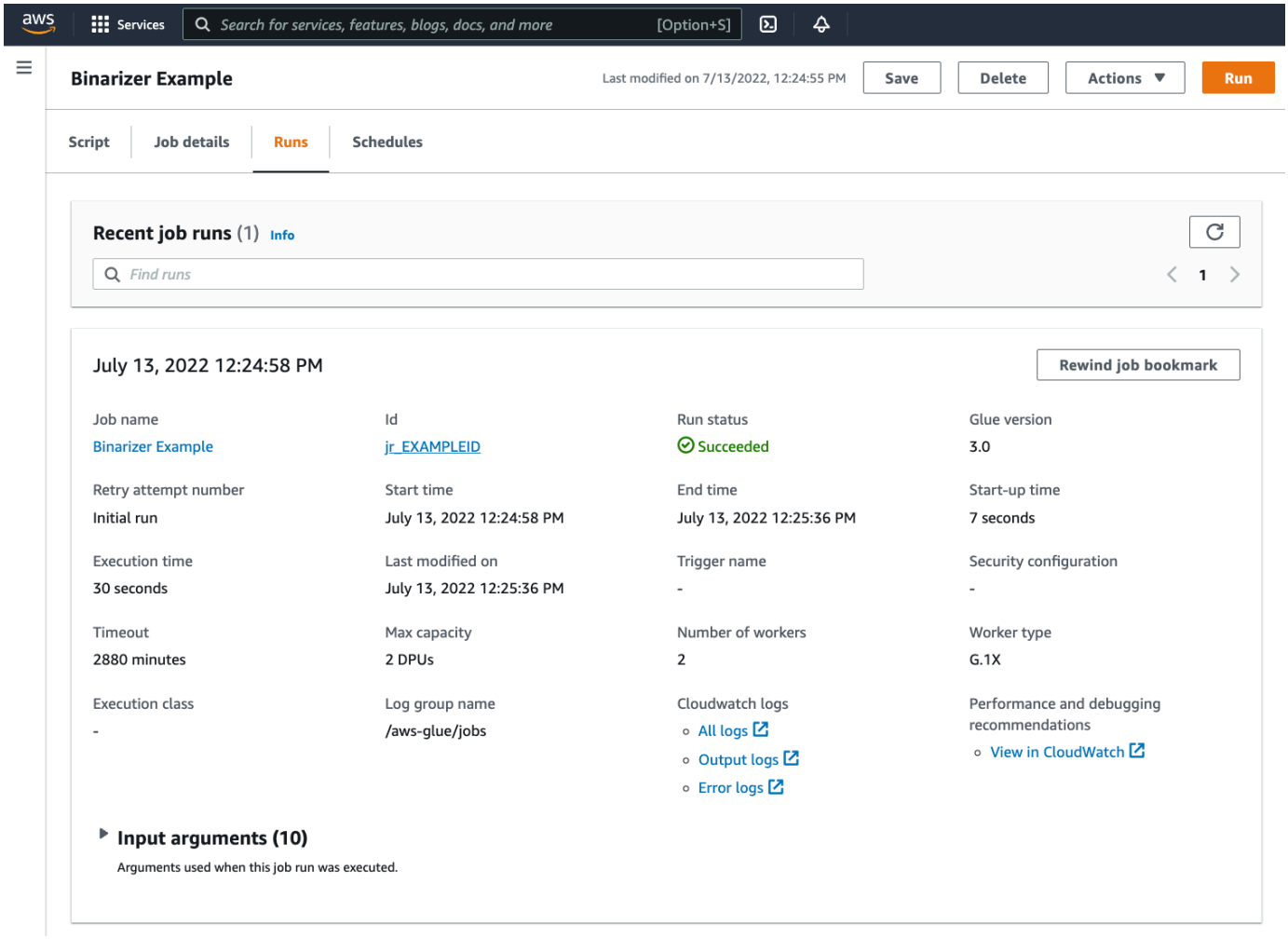
3. Ihr Browser navigiert zum Skript-Editor. Klicken Sie in der Kopfzeile auf die Registerkarte Auftragsdetails. Legen Sie den Namen und die IAM-Rolle fest. Anleitungen zu AWS Glue IAM-Rollen finden the section called “Einrichten von IAM-Berechtigungen” Sie unter.

   Optional – setzen Sie Angeforderte Anzahl der Worker auf 2 und die Anzahl der Wiederholungen auf 1. Diese Optionen sind nützlich, wenn Sie Produktionsaufträge ausführen, aber wenn Sie sie ablehnen, optimieren Sie Ihre Erfahrung beim Testen eines Features.

   Klicken Sie in der Titelleiste auf Save (Speichern) und dann auf Run (Ausführen)

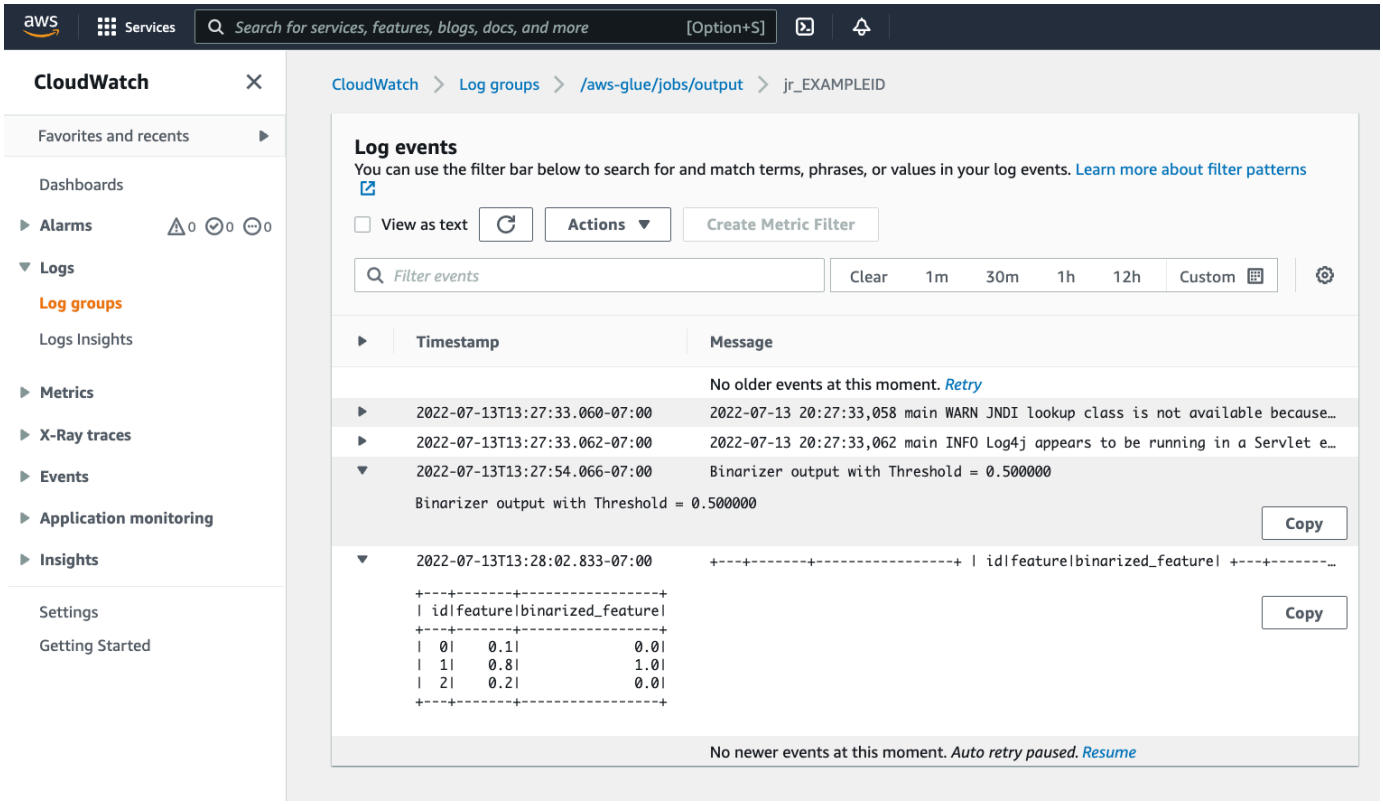


4. Navigieren Sie zur Registerkarte Ausführungen. Sie sehen ein Panel, das Ihrer Auftragsausführung entspricht. Warten Sie ein paar Minuten und die Seite sollte automatisch aktualisiert werden um Erfolgreich im Run status (Ausführungsstatus) anzuzeigen.



5. Sie sollten Ihre Ausgabe überprüfen, um sicherzustellen, dass das Spark-Skript wie beabsichtigt ausgeführt wurde. Dieses Apache Spark-Beispielskript sollte einen String in den Ausgabestream schreiben. Sie finden das, indem Sie zu Output logs (Ausgabeprotokolle) unter Cloudwatch-Protokolle im Panel für die erfolgreiche Auftragsausführung navigieren. Notieren Sie sich die Auftragsausführungs-ID, eine generierte ID unter dem ID-Label, das mit `jr_` beginnt.

   Dadurch wird die CloudWatch Konsole geöffnet, die so eingestellt ist, dass sie den Inhalt der AWS Glue Standard-Protokollgruppe visualisiert`/aws-glue/jobs/output`, gefiltert nach dem Inhalt der Protokollstreams für die Job-Ausführungs-ID. Jeder Worker hat einen Protokollstream generiert, der als Zeilen unter Protokollstreams dargestellt wird. Ein Worker hätte den angeforderten Code ausführen sollen. Sie müssen alle Protokollstream öffnen, um den richtigen Worker zu identifizieren. Sobald Sie den richtigen Worker gefunden haben, sollten Sie die Ausgabe des Skripts sehen, wie in der folgenden Abbildung zu sehen ist:



Gängige Verfahren für die Migration von Spark-Programmen

Bewerten Sie den Support der Spark-Version

AWS Glue Release-Versionen definieren die Version von Apache Spark und Python, die für den AWS Glue Job verfügbar sind. Sie finden unsere AWS Glue Versionen und was sie unterstützen unterthe section called “AWS Glue Versionen”. Möglicherweise müssen Sie Ihr Spark-Programm aktualisieren, um mit einer neueren Version von Spark kompatibel zu sein, um auf bestimmte AWS Glue -Features zuzugreifen.

Bibliotheken von Drittanbietern einschließen

Viele bestehende Spark-Programme werden Abhängigkeiten haben, sowohl von privaten als auch von öffentlichen Artefakten. AWS Glue unterstützt Abhängigkeiten im JAR-Stil für Scala-Aufträge sowie Wheel- und Quell-Pure-Python-Abhängigkeiten für Python-Aufträge.

Python – Hinweise zu Python-Abhängigkeiten finden Sie unter the section called “Python-Bibliotheken”

Allgemeine Python-Abhängigkeiten werden in der AWS Glue Umgebung bereitgestellt, einschließlich der häufig angeforderten Pandas-Bibliothek. Diese Abhängigkeiten sind in der AWS Glue -Version

2.0+ enthalten. Weitere Informationen zu bereitgestellten Modulen finden Sie unter the section called “Python-Module sind bereits in AWS Glue enthalten”. Wenn Sie einen Auftrag mit einer anderen Version einer standardmäßig enthaltenen Abhängigkeit bereitstellen müssen, können Sie `--additional-python-modules` verwenden. Informationen über Auftragsargumente finden Sie in the section called “Auftragsparameter”.

Sie können zusätzliche Python-Abhängigkeiten mit dem `--extra-py-files`-Auftragsargument liefern. Wenn Sie einen Job aus einem Spark-Programm migrieren, ist dieser Parameter eine gute Option, da er funktionell dem `--py-files` Flag in PySpark entspricht und denselben Einschränkungen unterliegt. Weitere Informationen zum Parameter `--extra-py-files` erhalten Sie unter the section called “Einschließlich Python-Dateien mit PySpark nativen Funktionen”.

Für neue Jobs können Sie Python-Abhängigkeiten mit dem `--additional-python-modules`-Auftragsargument verwalten. Die Verwendung dieses Arguments ermöglicht ein gründlicheres Abhängigkeitsmanagement. Dieser Parameter unterstützt Abhängigkeiten im Wheel-Stil, einschließlich solcher mit nativen Codebindungen, die mit Amazon Linux 2 kompatibel sind.

Scala

Sie können zusätzliche Scala-Abhängigkeiten mit dem `--extra-jars`-Auftragsargument liefern. Abhängigkeiten müssen in Amazon S3 gehostet werden und der Argumentwert sollte eine kommagetrennte Liste von Amazon S3-Pfaden ohne Leerzeichen sein. Möglicherweise fällt es Ihnen leichter, Ihre Konfiguration zu verwalten, indem Sie Ihre Abhängigkeiten neu bündeln, bevor Sie sie hosten und konfigurieren. AWS Glue JAR-Abhängigkeiten enthalten Java-Bytecode, der aus jeder JVM-Sprache generiert werden kann. Sie können andere JVM-Sprachen wie Java verwenden, um benutzerdefinierte Abhängigkeiten zu schreiben.

Anmeldeinformationen für Datenquellen verwalten

Bestehende Spark-Programme können mit einer komplexen oder benutzerdefinierten Konfiguration ausgestattet sein, um Daten aus ihren Datenquellen abzurufen. Gängige Authentifizierungsabläufe für Datenquellen werden von Verbindungen unterstützt. AWS Glue Weitere Informationen zu AWS Glue - Verbindungen finden Sie unter Herstellen einer Verbindung zu Daten.

AWS Glue Verbindungen ermöglichen es, Ihren Job auf zwei Arten von Datenspeichern mit einer Vielzahl von Arten von Datenspeichern zu verbinden: durch Methodenaufrufe an unsere Bibliotheken und durch das Einrichten der zusätzlichen Netzwerkverbindung in der AWS Konsole. Sie können das AWS SDK auch von Ihrem Job aus aufrufen, um Informationen aus einer Verbindung abzurufen.

Methodenaufrufe — AWS Glue Verbindungen sind eng in den AWS Glue Datenkatalog integriert, einen Dienst, der es Ihnen ermöglicht, Informationen über Ihre Datensätze zu kuratieren, und die verfügbaren Methoden zur Interaktion mit AWS Glue Verbindungen spiegeln dies wider. Wenn Sie über eine bestehende Authentifizierungskonfiguration verfügen, die Sie für JDBC-Verbindungen wiederverwenden möchten, können Sie auf Ihre AWS Glue Verbindungskonfiguration über die Methode auf der zugreifen. `extract_jdbc_conf` `GlueContext` Weitere Informationen finden Sie unter the section called “extract_jdbc_conf”

Konsolenkonfiguration — AWS Glue Jobs verwenden zugehörige AWS Glue Verbindungen, um Verbindungen zu Amazon VPC-Subnetzen zu konfigurieren. Wenn Sie Ihre Sicherheitsmaterialien direkt verwalten, müssen Sie in der AWS Konsole möglicherweise den `NETWORK` Typ Zusätzliche Netzwerkverbindung angeben, um das Routing zu konfigurieren. Weitere Informationen über die AWS Glue -Verbindungs-API finden Sie unter the section called “Verbindungen”

Wenn Ihre Spark-Programme über einen benutzerdefinierten oder ungewöhnlichen Authentifizierungsablauf verfügen, müssen Sie Ihre Sicherheitsmaterialien möglicherweise praxisnah verwalten. Wenn AWS Glue Verbindungen nicht gut zu passen scheinen, können Sie Sicherheitsmaterialien sicher in Secrets Manager hosten und über das Boto3 oder AWS SDK, die im Job bereitgestellt werden, darauf zugreifen.

Konfigurieren von Apache Spark

Komplexe Migrationen ändern häufig die Spark-Konfiguration, um ihre Workloads zu berücksichtigen. Moderne Versionen von Apache Spark ermöglichen die Einstellung der Laufzeitkonfiguration mit dem. `SparkSession` AWS Glue Es werden Jobs über 3.0 bereitgestellt`SparkSession`, die geändert werden können, um die Laufzeitkonfiguration festzulegen. Apache Sparkkonfiguration. Das Tuning von Spark ist komplex und garantiert AWS Glue nicht, dass die gesamte Spark-Konfiguration eingerichtet wird. Wenn Ihre Migration eine umfangreiche Konfiguration auf Spark-Ebene erfordert, wenden Sie sich an den Support.

Erstellen einer benutzerdefinierten Konfiguration

Migrierte Spark-Programme können so konzipiert sein, dass sie eine benutzerdefinierte Konfiguration erfordern. AWS Glue ermöglicht es, die Konfiguration über die Job-Argumente auf Job- und Job-Ausführungsebene festzulegen. Informationen über Auftragsargumente finden Sie in the section called “Auftragsparameter”. Sie können über unsere Bibliotheken auf Job-Argumente im Kontext eines Jobs zugreifen. AWS Glue bietet eine Hilfsfunktion, um eine konsistente Ansicht zwischen den Argumenten, die im Job gesetzt wurden, und den Argumenten, die während der Jobausführung

gesetzt wurden, bereitzustellen. Siehe the section called “getResolvedOptions” in Python und the section called “GlueArgParser” in Scala.

Migration von Java-Code

Wie in the section called “Bibliotheken von Drittanbietern” erklärt, können Ihre Abhängigkeiten Klassen enthalten, die von JVM-Sprachen wie Java oder Scala generiert wurden. Ihre Abhängigkeiten können eine `main`-Methode einschließen. Sie können eine `main` Methode in einer Abhängigkeit als Einstiegspunkt für einen AWS Glue Scala-Job verwenden. Auf diese Weise können Sie Ihre `main`-Methode in Java schreiben oder eine `main`-Methode auf Ihre eigenen Bibliotheksstandards gepackt wiederverwenden.

Um eine `main`-Methode aus einer Abhängigkeit zu verwenden, führen Sie Folgendes aus: Löschen Sie den Inhalt des Bearbeitungsbereichs, indem Sie das standardmäßige `GlueApp`-Objekt angeben. Geben Sie den vollqualifizierten Namen einer Klasse in einer Abhängigkeit als Auftragsargument mit dem Schlüssel `--class` an. Sie sollten nun einen Auftragsausführung auslösen können.

Sie können die Reihenfolge oder Struktur der AWS Glue Argumentübergaben an die Methode nicht konfigurieren. `main` Wenn Ihr vorhandener Code die eingestellte Konfiguration lesen muss AWS Glue, führt dies wahrscheinlich zu einer Inkompatibilität mit dem vorherigen Code. Wenn Sie `getResolvedOptions` verwenden, haben Sie ebenso keinen guten Ort, um diese Methode aufzurufen. Erwägen Sie, Ihre Abhängigkeit direkt von einer Hauptmethode aufzurufen, die von AWS Glue generiert wird. Das folgende AWS Glue ETL-Skript zeigt ein Beispiel dafür.

```
import com.amazonaws.services.glue.util.GlueArgParser

object GlueApp {
  def main(sysArgs: Array[String]) {
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)

    // Invoke static method from JAR. Pass some sample arguments as a String[], one
 defined inline and one taken from the job arguments, using getResolvedOptions
    com.mycompany.myproject.MyClass.myStaticPublicMethod(Array("string parameter1",
 args("JOB_NAME")))

    // Alternatively, invoke a non-static public method.
    (new com.mycompany.myproject.MyClass).someMethod()
  }
}
```

Arbeiten mit Ray Jobs in AWS Glue

Dieser Abschnitt enthält Informationen zur Verwendung von AWS Glue For Ray-Jobs. Weitere Informationen AWS Glue zum Schreiben von Ray-Skripten finden Sie im the section called “AWS Glue für Ray” Abschnitt.

Themen

- Erste Schritte mit AWS Glue for Ray
- Unterstützte Ray-Laufzeitumgebungen
- Abrechnung für Worker in Ray-Aufträgen
- Verwendung von Auftragsparametern in Ray-Aufträgen
- Überwachung von Ray-Aufträgen mit Metriken

Erste Schritte mit AWS Glue for Ray

Für die Arbeit mit AWS Glue for Ray verwenden Sie dieselben AWS Glue Jobs und interaktiven Sessions wie AWS Glue für Spark. AWS Glue Jobs sind für die Ausführung desselben Skripts in wiederkehrenden Abständen konzipiert, während interaktive Sitzungen darauf ausgelegt sind, dass Sie Codefragmente sequentiell für dieselben bereitgestellten Ressourcen ausführen können.

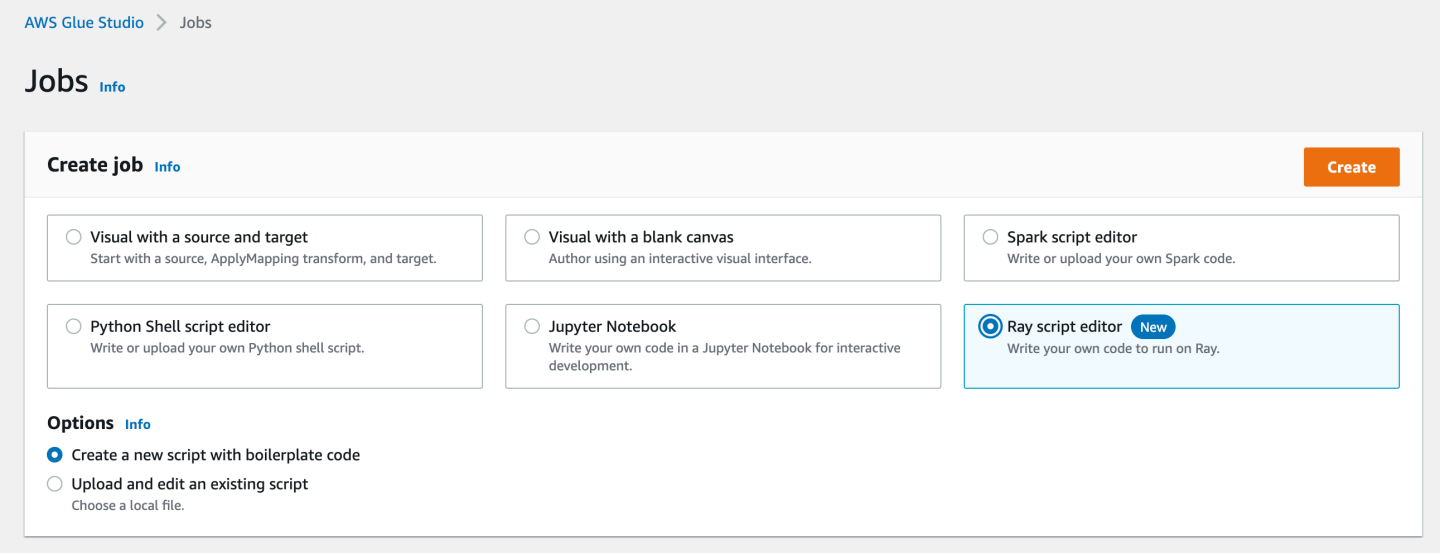
AWS Glue ETL und Ray unterscheiden sich darunter, sodass Sie in Ihrem Skript Zugriff auf verschiedene Tools, Funktionen und Konfigurationen haben. Als neues Berechnungsframework, das von verwaltet wird AWS Glue, hat Ray eine andere Architektur und verwendet ein anderes Vokabular, um zu beschreiben, was es tut. Weitere Informationen finden Sie in den Architektur-Whitepapers in der Ray-Dokumentation.

> **Note**
>
> AWS Glue for Ray ist in den Ländern USA Ost (Nord-Virginia), USA Ost (Ohio), USA West (Oregon), Asien-Pazifik (Tokio) und Europa (Irland) verfügbar.

Ray-Jobs in der AWS Glue Studio Konsole

Auf der Seite Jobs in der AWS Glue Studio Konsole können Sie eine neue Option auswählen, wenn Sie einen Job im AWS Glue Studio— Ray-Skripteditor erstellen. Wählen Sie diese Option, um einen

Ray-Auftrag in der Konsole zu erstellen. Weitere Informationen zu Aufträgen und deren Verwendung finden Sie unter Erstellung visueller ETL-Jobs.



Ray-Jobs im AWS CLI und SDK

Ray-Jobs im AWS CLI verwenden dieselben SDK-Aktionen und -Parameter wie andere Jobs. AWS Glue for Ray führt neue Werte für bestimmte Parameter ein. Weitere Informationen zur Auftrags-API finden Sie unter the section called “Aufträge”.

Unterstützte Ray-Laufzeitumgebungen

Bei Spark-Aufträgen ermittelt `GlueVersion` die Versionen von Apache Spark und Python, die in jedem AWS Glue für Spark-Aufträge verfügbar sind. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird. Dies ist nicht die Art und Weise, wie Ray-Laufzeitumgebungen konfiguriert sind.

Für Ray-Aufträge sollten Sie `GlueVersion` auf `4.0` oder höher festlegen. Welche Versionen von Ray, Python und weiteren Bibliotheken in Ihrem Ray-Auftrag verfügbar sind, wird jedoch durch das `Runtime`-Feld in der Auftragsdefinition bestimmt.

Die `Ray2.4` Laufzeitumgebung steht nach der Veröffentlichung mindestens 6 Monate lang zur Verfügung. Da sich Ray schnell weiterentwickelt, können Sie Aktualisierungen und Verbesserungen von Ray über zukünftige Versionen der Laufzeitumgebung einbinden.

Zulässige Werte: `Ray2.4`

Wert der Laufzeit	Ray- und Python-Versionen
`Ray2.4`(für AWS Glue 4.0+)	Ray 2.4.0 Python 3.9

Zusätzliche Informationen

- Versionshinweise, die den Veröffentlichungen von Ray AWS Glue beiliegen, finden Sie unterthe section called “AWS Glue Versionen”.
- Informationen zu Python-Bibliotheken, die in einer Laufzeitumgebung bereitgestellt werden, finden Sie unter the section called “Mit Ray-Aufträgen bereitgestellte Module”.

Abrechnung für Worker in Ray-Aufträgen

AWS Glue führt Ray-Jobs auf neuen Graviton-basierten EC2 Worker-Typen aus, die nur für Ray-Jobs verfügbar sind. Um diese Worker angemessen für die Workloads bereitzustellen, für die Ray entwickelt wurde, stellen wir für die meisten Worker ein anderes Verhältnis von Rechenressourcen zu Speicherressourcen bereit. Um diese Ressourcen zu berücksichtigen, verwenden wir die speicheroptimierte Datenverarbeitungseinheit (M-DPU) anstelle der Standard-Datenverarbeitungseinheit (DPU).

- Eine M-DPU entspricht 4 V CPUs und 32 GB Arbeitsspeicher.
- Eine DPU entspricht 4 V CPUs und 16 GB Speicher. DPUs werden verwendet, um Ressourcen in AWS Glue Spark-Jobs und entsprechenden Mitarbeitern zu berücksichtigen.

Ray-Aufträge haben derzeit Zugriff auf einen Worker-Typ, `Z.2X`. Der `Z.2X` Worker entspricht 2 M-DPUs (8 VCPUs, 64 GB Arbeitsspeicher) und verfügt über 128 GB Festplattenspeicher. Ein `Z.2X`-Computer stellt 8 Ray-Worker bereit (einen pro vCPU).

Die Anzahl der M-DPUs , die Sie gleichzeitig in einem Konto verwenden können, unterliegt einem Servicekontingent. Weitere Informationen zu Ihren AWS Glue Kontolimits finden Sie unter AWS Glue Endpunkte und Kontingente.

Die Anzahl der Worker-Knoten, die einem Ray-Auftrag zur Verfügung stehen, geben Sie mit `--number-of-workers (NumberOfWorkers)` in der Auftragsdefinition an. Weitere Informationen zu Ray-Werten in der Auftrags-API finden Sie unter the section called “Aufträge”.

Mit dem `--min-workers`-Auftragsparameter können Sie außerdem eine Mindestanzahl von Workern angeben, die ein Ray-Auftrag zuweisen muss. Informationen zu Auftragsparametern finden Sie unter the section called “Referenz”.

Verwendung von Auftragsparametern in Ray-Aufträgen

Sie legen Argumente für AWS Glue Ray-Jobs genauso fest wie Argumente AWS Glue für Spark-Jobs. Weitere Informationen zur AWS Glue API finden Sie unterthe section called “Aufträge”. Sie können AWS Glue Ray-Jobs mit verschiedenen Argumenten konfigurieren, die in dieser Referenz aufgeführt sind. Sie können auch Ihre eigenen Argumente angeben.

Sie können einen Auftrag über die Konsole auf der Registerkarte Job details (Auftragsdetails) unter der Überschrift Job Parameters (Auftragsparameter) konfigurieren. Sie können einen Job auch über die Einstellung AWS CLI by für einen Job oder für die Einstellung `DefaultArguments` für einen Job, `Arguments` der ausgeführt wird, konfigurieren. Standardargumente und Auftragsparameter bleiben bei mehreren Ausführungen des Auftrags erhalten.

Im Folgenden wird die Syntax zum Ausführen eines Auftrags mit `--arguments` für das Festlegen eines speziellen Parameters gezeigt.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py",--test-environment="true"'
```

Nachdem Sie die Argumente festgelegt haben, können Sie von Ihrem Ray-Auftrag aus über Umgebungsvariablen auf die Auftragsparameter zugreifen. Auf diese Weise können Sie Ihren Auftrag für jede Ausführung konfigurieren. Der Name der Umgebungsvariablen ist der Name des Auftragsarguments ohne das `---`Präfix.

Im vorherigen Beispiel würden die Variablennamen beispielsweise `scriptLocation` und `test-environment` lauten. Anschließend würden Sie das Argument über die in der Standardbibliothek verfügbaren Methoden abrufen: `test_environment = os.environ.get('test-environment')`. Weitere Informationen zum Zugriff auf Umgebungsvariablen mit Python finden Sie unter OS-Modul in der Python-Dokumentation.

Konfigurieren, wie Ray-Aufträge Protokolle erzeugen

Standardmäßig generieren Ray-Jobs Protokolle und Metriken, die an CloudWatch Amazon S3 gesendet werden. Sie können den `--logging_configuration`-Parameter verwenden, um zu ändern, wie Protokolle erzeugt werden. Derzeit können Sie damit verhindern, dass Ray-Aufträge verschiedene Arten von Protokollen erzeugen. Dieser Parameter nimmt ein JSON-Objekt auf, dessen Schlüssel den Protokollen/Verhaltensweisen entsprechen, die Sie ändern möchten. Es unterstützt die folgenden Schlüssel:

- `CLOUDWATCH_METRICS`— Konfiguriert eine Reihe von CloudWatch Metriken, die zur Visualisierung des Arbeitszustands verwendet werden können. Weitere Informationen zu den Metriken finden Sie unter the section called “Auftragsmetriken von Ray”.
- `CLOUDWATCH_LOGS`— Konfiguriert CloudWatch Protokolle, die Ray-Informationen über den Status der Ausführung des Jobs auf Anwendungsebene enthalten. Weitere Informationen über Protokolle finden Sie unter the section called “Fehlerbehebung bei Ray-Fehlern”.
- `S3`— Konfiguriert, was in Amazon S3 AWS Glue geschrieben wird, hauptsächlich ähnliche Informationen wie CloudWatch Logs, aber als Dateien und nicht als Log-Streams.

Um ein Ray-Protokollierungsverhalten zu deaktivieren, geben Sie den Wert `{\"IS_ENABLED \": \"False\"}` an. Um beispielsweise CloudWatch Metriken und CloudWatch Protokolle zu deaktivieren, geben Sie die folgende Konfiguration an:

```
"--logging_configuration": "{\"CLOUDWATCH_METRICS\": {\"IS_ENABLED\": \"False\"},
 \"CLOUDWATCH_LOGS\": {\"IS_ENABLED\": \"False\"}}"
```

Referenz

Ray-Aufträge erkennen die folgenden Argumentnamen, die Sie zum Einrichten der Skriptumgebung für Ihre Ray-Aufträge und Auftragsausführungen verwenden können:

- `--logging_configuration` – Wird verwendet, um die Erzeugung verschiedener Protokolle zu stoppen, die von Ray-Aufträgen erstellt werden. Diese Protokolle werden standardmäßig für alle Ray-Aufträge erstellt. Format: String-escaped JSON-Objekt. Weitere Informationen finden Sie unter the section called “Konfigurieren, wie Ray-Aufträge Protokolle erzeugen”.
- `--min-workers` – Die Mindestanzahl von Worker-Knoten, die einem Ray-Auftrag zugewiesen werden. Ein Worker-Knoten kann mehrere Replikate ausführen, eines pro virtueller CPU. Format: Ganzzahl. Minimum: 0. Maximum: Der in `--number-of-workers (NumberOfWorkers)`

angegebene Wert in der Auftragsdefinition. Weitere Informationen zur Abrechnung von Worker-Knoten finden Sie unter the section called “Abrechnung für Worker in Ray-Aufträgen”.

- `--object_spilling_config`— AWS Glue denn Ray unterstützt die Verwendung von Amazon S3 als Möglichkeit, den für Rays Objektspeicher verfügbaren Speicherplatz zu erweitern. Um dieses Verhalten zu aktivieren, können Sie Ray mit diesem Parameter ein JSON-Konfigurationsobjekt zur Verfügung stellen, das Objekte ausgibt. Weitere Informationen zur Konfiguration der Ray-Objekt-Ausgabe finden Sie unter Objekt-Ausgabe in der Ray-Dokumentation. Format: JSON-Objekt.

  AWS Glue for Ray unterstützt nur die gleichzeitige Übertragung auf die Festplatte oder die gleichzeitige Übertragung auf Amazon S3. Sie können mehrere Standorte für die Ausgabe angeben, sofern diese Einschränkung eingehalten wird. Wenn die Ausgabe zu Amazon S3 erfolgt, müssen Sie Ihrem Auftrag für diesen Bucket auch IAM-Berechtigungen hinzufügen.

  Wenn Sie ein JSON-Objekt als Konfiguration mit der CLI bereitstellen, müssen Sie es als Zeichenfolge bereitstellen, wobei die Zeichenfolge des JSON-Objekts mit Escape-Zeichen versehen ist. Ein Zeichenfolgenwert zum Übertragen auf einen Amazon S3-Pfad würde beispielsweise wie folgt aussehen: `"{\"type\": \"smart_open\", \"params\": {\"uri\":\"s3path\"}}"`. Geben Sie diesen Parameter als JSON-Objekt ohne zusätzliche Formatierung an. AWS Glue Studio
- `--object_store_memory_head` – Der Speicher, der dem Plasma-Objektspeicher auf dem Ray-Head-Knoten zugewiesen ist. Diese Instance führt Cluster-Management-Services sowie Worker-Replikate aus. Der Wert stellt den Prozentsatz des freien Speichers auf der Instance nach einem Warmstart dar. Sie verwenden diesen Parameter, um speicherintensive Workloads zu optimieren – die Standardeinstellungen sind für die meisten Anwendungsfälle akzeptabel. Format: positive Ganzzahl. Minimum: 1. Maximum: 100.

  Weitere Informationen zu Plasma finden Sie unter The Plasma In-Memory Object Store in der Ray-Dokumentation.
- `--object_store_memory_worker` – Der Speicher, der dem Plasma-Objektspeicher auf den Ray-Worker-Knoten zugewiesen ist. Diese Instances führen nur Worker-Replikate aus. Der Wert stellt den Prozentsatz des freien Speichers auf der Instance nach einem Warmstart dar. Dieser Parameter wird zum Optimieren speicherintensiver Workloads verwendet – die Standardwerte sind für die meisten Anwendungsfälle akzeptabel. Format: positive Ganzzahl. Minimum: 1. Maximum: 100.

Weitere Informationen zu Plasma finden Sie unter The Plasma In-Memory Object Store in der Ray-Dokumentation.

- `--pip-install` – Eine Reihe von zu installierenden Python-Paketen. Mit diesem Argument können Sie Pakete von PyPI installieren. Format: durch Kommas getrennte Liste.

  Ein PyPI-Paketeintrag liegt im Format `package==version` vor und enthält den PyPI-Namen und die Version Ihres Zielpakets. Einträge verwenden den Python-Versionsabgleich, um das Paket und die Version abzugleichen, wie z. B. ==, nicht das einzelne Gleichheitszeichen =. Es gibt andere Operatoren für den Versionsabgleich. Weitere Informationen finden Sie unter PEP 440 auf der Python-Website. Sie können auch benutzerdefinierte Module mit `--s3-py-modules` bereitstellen.
- `--s3-py-modules` – Eine Reihe von Amazon-S3-Pfaden, die Python-Modulverteilungen hosten. Format: durch Kommas getrennte Liste.

  Sie können dies verwenden, um Ihre eigenen Module an Ihren Ray-Auftrag zu verteilen. Sie können mit `--pip-install` auch Module von PyPI bereitstellen. Im Gegensatz zu AWS Glue ETL werden benutzerdefinierte Module nicht über Pip eingerichtet, sondern zur Verteilung an Ray übergeben. Weitere Informationen finden Sie unter the section called “Zusätzliche Python-Module für Ray-Aufträge”.
- `--working-dir` – Ein Pfad zu einer in Amazon S3 gehosteten ZIP-Datei, die Dateien enthält, die an alle Knoten verteilt werden sollen, auf denen Ihr Ray-Auftrag ausgeführt wird. Format: Zeichenfolge. Weitere Informationen finden Sie unter the section called “Bereitstellen von Dateien für Ihren Ray-Auftrag”.

Überwachung von Ray-Aufträgen mit Metriken

Sie können Ray-Jobs mit AWS Glue Studio Amazon überwachen CloudWatch. CloudWatch sammelt und verarbeitet Rohmetriken von AWS Glue Ray und stellt sie so für Analysen zur Verfügung. Diese Metriken werden in der AWS Glue Studio Konsole visualisiert, sodass Sie Ihren Job während der Ausführung überwachen können.

Einen allgemeinen Überblick über die Überwachung finden Sie AWS Glue unterthe section called “ CloudWatch Metriken verwenden”. Einen allgemeinen Überblick über die Verwendung von CloudWatch Metriken, die von veröffentlicht wurden AWS Glue, finden Sie unterthe section called “Überwachung mit CloudWatch”.

Ray-Jobs in der AWS Glue Konsole überwachen

Auf der Detailseite für einen Joblauf unter dem Abschnitt Ausführungsdetails können Sie sich vorgefertigte aggregierte Diagramme ansehen, die Ihre verfügbaren Job-Metriken visualisieren. AWS Glue Studio sendet Job-Metriken CloudWatch für jeden Job-Lauf an. Damit können Sie ein Profil Ihres Clusters und Ihrer Aufgaben erstellen und auf detaillierte Informationen zu jedem Knoten zugreifen.

Weitere Informationen zu verfügbaren Metrikdiagrammen finden Sie unter the section called “Amazon CloudWatch Metriken für einen Ray-Joblauf anzeigen”.

Überblick über die Ray-Job-Metriken in CloudWatch

Wir veröffentlichen Ray-Metriken, wenn die detaillierte Überwachung aktiviert ist CloudWatch. Metriken werden im `Glue/Ray` CloudWatch Namespace veröffentlicht.

- Instance-Metriken

  Wir veröffentlichen Metriken über die CPU-, Speicher- und Festplattenauslastung von Instances, die einem Auftrag zugewiesen sind. Diese Metriken werden anhand von Features wie `ExecutorId`, `ExecutorType` und `host` identifiziert. Diese Metriken sind eine Teilmenge der Standardmetriken für CloudWatch Linux-Agenten. Informationen zu den Namen und Funktionen von Metriken finden Sie in der CloudWatch Dokumentation. Weitere Informationen finden Sie unter Vom CloudWatch Agenten gesammelte Metriken.

- Ray-Cluster-Metriken

  Wir leiten Metriken von den Ray-Prozessen, die Ihr Skript ausführen, an diesen Namespace weiter und stellen Ihnen dann die für Sie wichtigsten Metriken zur Verfügung. Die verfügbaren Metriken können sich je nach Ray-Version unterscheiden. Weitere Informationen dazu, welche Ray-Version Ihr Auftrag ausführt, finden Sie unter the section called “AWS Glue Versionen”.

  Ray erfasst Metriken auf Instance-Ebene. Es stellt auch Metriken für Aufgaben und den Cluster bereit. Weitere Informationen zur zugrunde liegenden Metrikstrategie von Ray finden Sie unter Metriken in der Ray-Dokumentation.

> **Note**
>
> Wir veröffentlichen keine Ray-Metriken im `Glue/Job Metrics/`-Namespace, der nur für AWS Glue -ETL-Aufträge verwendet wird.

Konfiguration von Jobeigenschaften für Python-Shell-Jobs in AWS Glue

Sie können einen Python-Shell-Job verwenden, um Python-Skripte als Shell auszuführen in AWS Glue. Mit einem Python-Shell-Job können Sie Skripte ausführen, die mit Python 3.6 oder Python 3.9 kompatibel sind.

> **Note**
>
> Die Support für Pyshell v3.6 endet am 1. März 2026. Informationen zur Migration Ihrer Workloads finden Sie unter Migration von AWS Glue Python-Shell-Jobs. Wenn Sie mit der Python-Shell v3.9 fortfahren möchten, sieheMigration von Python-Shell 3.6 auf Python-Shell 3.9.

Themen

- Einschränkungen
- Definieren von Auftragseigenschaften für Python-Shell-Aufträge
- Für Python-Shell-Aufträge unterstützte Bibliotheken
- Bereitstellen Ihrer eigenen Python-Bibliothek
- Verwendung AWS CloudFormation mit Python-Shell-Jobs in AWS Glue
- Migration von Python-Shell 3.6 auf Python-Shell 3.9
- Migrieren Sie von AWS Glue Python-Shell-Jobs

Einschränkungen

Beachten Sie die folgenden Einschränkungen von Python-Shell-Aufträgen:

- Sie können für Python-Shell-Aufträge keine Auftragslesezeichen verwenden.

- In Python 3.9+ können Sie keine Python-Bibliotheken als `.egg` Dateien packen. Nutzen Sie stattdessen `.whl`.
- Die `--extra-files`-Option kann aufgrund einer Beschränkung für temporäre Kopien von S3-Daten nicht verwendet werden.

Definieren von Auftragseigenschaften für Python-Shell-Aufträge

In diesen Abschnitten wird die Definition von Jobeigenschaften beschrieben in AWS Glue Studio, oder mit der AWS CLI.

AWS Glue Studio

Wenn Sie Ihren Python-Shell-Job definieren in AWS Glue Studio, geben Sie einige der folgenden Eigenschaften an:

IAM-Rolle

Geben Sie die AWS Identity and Access Management (IAM-) Rolle an, die für die Autorisierung von Ressourcen verwendet wird, die für die Ausführung des Jobs und den Zugriff auf Datenspeicher verwendet werden. Weitere Informationen zu Berechtigungen für die Ausführung von Jobs finden Sie in AWS Glue, finden Sie unter Identitäts- und Zugriffsmanagement für AWS Glue.

Typ

Wählen Sie Python shell (Python-Shell) aus, um ein Python-Skript mit dem Auftragsbefehl `pythonshell` auszuführen.

Python-Version

Wählen Sie die Python-Version aus. Die Standardeinstellung ist Python 3.9. Gültige Versionen sind Python 3.6 und Python 3.9.

Allgemeine Analyse-Bibliotheken laden (empfohlen)

Wählen Sie diese Option, um allgemeine Bibliotheken für Python 3.9 in die Python-Shell aufzunehmen.

Wenn Ihre Bibliotheken entweder benutzerdefiniert sind oder mit den vorinstallierten in Konflikt stehen, können Sie sich dafür entscheiden, keine gängigen Bibliotheken zu installieren. Sie können jedoch neben den üblichen Bibliotheken weitere Bibliotheken installieren.

Wenn Sie diese Option auswählen, wird die `library-set`-Option auf `analytics` gesetzt. Wenn Sie diese Option abwählen, wird die `library-set`-Option auf none gesetzt.

Name der Skriptdatei und Skriptpfad

Der Code im Skript definiert die prozedurale Logik Ihres Auftrags. Geben Sie den Skript-Namen und den Speicherort in Amazon Simple Storage Service (Amazon S3) an. Vergewissern Sie sich, dass sich keine Datei mit dem Namen des Skriptverzeichnisses im Pfad befindet. Weitere Informationen über die Verwendung von Skripts finden Sie unter AWS Glue Leitfaden zur Programmierung.

Script

Der Code im Skript definiert die prozedurale Logik Ihres Auftrags. Sie können das Skript in Python 3.6 oder Python 3.9 codieren. Sie können ein Skript bearbeiten in AWS Glue Studio.

Datenverarbeitungseinheiten

Die maximale Anzahl von AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie unter .AWS Glue Preisgestaltung.

Sie können den Wert auf 0,0625 oder 1 festlegen. Der Standardwert ist 0.0625. In beiden Fällen beträgt die lokale Festplatte für die Instance 20 GB.

CLI

Sie können auch einen Python-Shell-Job mit dem erstellen AWS CLI, wie im folgenden Beispiel.

```
aws glue create-job --name python-job-cli --role Glue_DefaultRole
    --command '{"Name" :  "pythonshell", "PythonVersion": "3.9", "ScriptLocation" :
"s3://amzn-s3-demo-bucket/scriptname.py"}'
    --max-capacity 0.0625
```

> **Note**
>
> Sie müssen die Version von nicht angeben AWS Glue da der Parameter `--glue-version` nicht gilt für AWS Glue Shell-Jobs. Jede angegebene Version wird ignoriert.

Jobs, die Sie mit der AWS CLI Standardeinstellung Python 3 erstellen. Gültige Python-Versionen sind 3 (entsprechend 3.6) und 3.9. Um Python 3.6 anzugeben, fügen Sie dieses Tupel dem `--command`-Parameter hinzu: `"PythonVersion":"3"`

Um Python 3.9 anzugeben, fügen Sie dieses Tupel dem `--command`-Parameter hinzu: `"PythonVersion":"3.9"`

Um die maximale Kapazität festzulegen, die von einem Python-Shell-Auftrag genutzt werden darf, geben Sie den Parameter `--max-capacity` an. Der Parameter `--allocated-capacity` kann für Python-Shell-Aufträge nicht verwendet werden.

Für Python-Shell-Aufträge unterstützte Bibliotheken

In der Python-Shell mit Python 3.9 können Sie den Bibliothekssatz auswählen, um vorgefertigte Bibliothekssätze für Ihre Anforderungen zu verwenden. Sie können die `library-set`-Option verwenden, um den Bibliothekssatz auszuwählen. Gültige Werte sind `analytics` und `none`.

Die Umgebung für die Ausführung von Python-Shell-Aufträgen unterstützt die folgenden Bibliotheken:

Python-Version	Python 3.6	Python 3.9	
Bibliothek-Set	N/A	Analytik	Keine
avro		1.11.0	
awscli	116.242	1,23,5	1,23,5
awswrangler		2.15,1	
botocore	1.12,232	1,24,21	1,23,5
boto3	1,9,203	1,21,21	
elasticsearch		8.2.0	
numpy	1,16.2	1.22.3	
pandas	0,24,2	1.4.2	
psycopg2		2.9.3	

Python-Version	Python 3.6	Python 3.9	
pyathena		2.5.3	
PyGreSQL	5.0.6		
PyMySQL		1.0.2	
pyodbc		4.0.32	
pyorc		0.6.0	
redshift-connector		2.0.907	
Anforderungen	2.22.0	2,27,1	
Scikit-learn	0,20,3	1.0.2	
scipy	1.2.1	1.8.0	
SQLAlchemy		1,4,36	
s3fs		2022,3,0	

Für wissenschaftliches Computing können Sie die Bibliothek `NumPy` in einer Python-Shell-Aufgabe verwenden. Weitere Informationen finden Sie unter NumPy. Das folgende Beispiel zeigt ein NumPy Skript, das in einem Python-Shell-Job verwendet werden kann. Das Skript gibt „Hello World“ und die Ergebnisse mehrerer mathematischer Berechnungen aus.

```
import numpy as np
print("Hello world")

a = np.array([20,30,40,50])
print(a)

b = np.arange( 4 )

print(b)

c = a-b
```

```
print(c)

d = b**2

print(d)
```

Bereitstellen Ihrer eigenen Python-Bibliothek

Verwenden von PIP

Mit Python-Shell, die Python 3.9 verwendet, können Sie zusätzliche Python-Module oder verschiedene Versionen auf der Auftragsebene bereitstellen. Sie können die Option „`--additional-python-modules`“ mit verschiedenen kommagetrennten Python-Modulen verwenden, um ein neues Modul hinzuzufügen oder die Version eines vorhandenen Moduls zu ändern. Sie können benutzerdefinierte Python-Module, die auf Amazon S3 gehostet werden, nicht mit diesem Parameter bereitstellen, wenn Sie Python-Shell-Aufträge verwenden.

Verwenden Sie beispielsweise zum Aktualisieren oder Hinzufügen eines neuen `scikit-learn`-Moduls den folgenden Schlüssel und Wert: `"--additional-python-modules", "scikit-learn==0.21.3"`.

AWS Glue verwendet den Python Package Installer (pip3), um die zusätzlichen Module zu installieren. Sie können zusätzliche pip3-Optionen innerhalb des `--additional-python-modules`-Werts übergeben. Beispiel, `"scikit-learn==0.21.3 -i https://pypi.python.org/simple/"`. Es gelten alle Inkompatibilitäten oder Einschränkungen von pip3.

> **Note**
>
> Um Inkompatibilitäten in Zukunft zu vermeiden, empfehlen wir die Verwendung von Bibliotheken, die für Python 3.9 erstellt wurden.

Verwenden einer Egg- oder Whl-Datei

Möglicherweise verfügen Sie bereits über eine oder mehrere, als `.egg`- oder `.whl`-Datei verpackte Python-Bibliotheken. Wenn dies der Fall ist, können Sie diese für Ihre Aufgabe angeben, indem Sie die AWS Command Line Interface (AWS CLI) unter dem Flag „`--extra-py-files`“ verwenden wie im folgenden Beispiel gezeigt.

```
aws glue create-job --name python-redshift-test-cli --role role --command '{"Name" :
 "pythonshell", "ScriptLocation" : "s3://MyBucket/python/library/redshift_test.py"}'
     --connections Connections=connection-name --default-arguments '{"--extra-py-
files" : ["s3://amzn-s3-demo-bucket/EGG-FILE", "s3://amzn-s3-demo-bucket/WHEEL-FILE"]}'
```

Wenn Sie sich nicht sicher sind, wie Sie eine `.egg`- oder `.whl`-Datei aus einer Python-Bibliothek erstellen, führen Sie die folgenden Schritte aus. Dieses Beispiel gilt für macOS, Linux und Windows Subsystem for Linux (WSL).

So erstellen Sie eine Python-EGG- oder -WHL-Datei

1. Erstellen Sie einen Amazon-Redshift-Cluster in einer Virtual Private Cloud (VPC) und fügen Sie einer Tabelle einige Daten hinzu.
2. Erstellen Sie ein AWS Glue Verbindung für die VPC-SecurityGroup-Subnet Kombination, mit der Sie den Cluster erstellt haben. Testen Sie, ob die Verbindung erfolgreich ist.
3. Erstellen Sie ein Verzeichnis mit dem Namen `redshift_example` und eine Datei mit dem Namen `setup.py`. Fügen Sie folgenden Code in ei `setup.py`.

   ```
   from setuptools import setup

   setup(
       name="redshift_module",
       version="0.1",
       packages=['redshift_module']
   )
   ```
4. Erstellen Sie im Verzeichnis `redshift_example` das Verzeichnis `redshift_module`. Erstellen Sie im Verzeichnis `redshift_module` die Dateien `__init__.py` und `pygresql_redshift_common.py`.
5. Lassen Sie die Datei `__init__.py` leer. Fügen Sie in `pygresql_redshift_common.py` folgenden Code ein. Ersetzen Sie*`port`*, *`db_name`*`user`, und *`password_for_user`* durch spezifische Details für Ihren Amazon Redshift Redshift-Cluster. *`table_name`*Ersetzen Sie durch den Namen der Tabelle in Amazon Redshift.

   ```
   import pg


   def get_connection(host):
       rs_conn_string = "host=%s port=%s dbname=%s user=%s password=%s" % (
   ```

```
        host, port, db_name, user, password_for_user)

    rs_conn = pg.connect(dbname=rs_conn_string)
    rs_conn.query("set statement_timeout = 1200000")
    return rs_conn


def query(con):
    statement = "Select * from table_name;"
    res = con.query(statement)
    return res
```

6. Wechseln Sie zum Verzeichnis `redshift_example`, sofern Sie dieses noch nicht aufgerufen haben.
7. Führen Sie eine der folgenden Aktionen aus:

   - Führen Sie den folgenden Befehl aus, um eine `.egg`-Datei zu erstellen.

     ```
     python setup.py bdist_egg
     ```

   - Führen Sie den folgenden Befehl aus, um eine `.whl`-Datei zu erstellen.

     ```
     python setup.py bdist_wheel
     ```

8. Installieren Sie die für den vorangehenden Befehl erforderlichen Abhängigkeiten.
9. Der Befehl erstellt eine Datei im Verzeichnis `dist`.

   - Wenn Sie eine EGG-Datei erstellt haben, hat diese den Namen `redshift_module-0.1-py2.7.egg`.
   - Wenn Sie eine Wheel-Datei erstellt haben, hat diese den Namen `redshift_module-0.1-py2.7-none-any.whl`.

   Laden Sie diese Datei in Amazon S3 hoch.

   In diesem Beispiel ist der Pfad der hochgeladenen Datei *s3://amzn-s3-demo-bucket/EGG-FILE* oder *s3://amzn-s3-demo-bucket/WHEEL-FILE*.
10. Erstellen Sie eine Python-Datei, die als Skript für die verwendet werden soll AWS Glue Job und fügen Sie der Datei den folgenden Code hinzu.

```
from redshift_module import pygresql_redshift_common as rs_common

con1 = rs_common.get_connection(redshift_endpoint)
res = rs_common.query(con1)

print "Rows in the table cities are: "

print res
```

11. Laden Sie die vorangehende Datei in Amazon S3 hoch. In diesem Beispiel ist der Pfad der hochgeladenen Datei *s3://amzn-s3-demo-bucket/scriptname.py*.
12. Erstellen Sie einen Python-Shell-Auftrag, der dieses Skript verwendet. Auf dem AWS Glue In der Konsole geben Sie auf der Seite mit den Auftragseigenschaften den Pfad zur `.egg/.whl` Datei im Feld Python-Bibliothekspfad an. Wenn es mehrere `.egg/.whl`- und Python-Dateien gibt, geben Sie in diesem Feld eine durch Komma getrennte Liste ein.

    Beim Ändern oder Umbenennen von `.egg`-Dateien müssen die Dateinamen die Standardnamen verwenden, die mit dem Befehl „python setup.py bdist_egg“ generiert werden, oder die Benennungskonventionen des Python-Moduls einhalten. Weitere Informationen finden Sie in den Vorgaben für Python-Code.

    Erstellen Sie mithilfe von einen Job mit einem Befehl, wie im folgenden Beispiel. AWS CLI

    ```
    aws glue create-job --name python-redshift-test-cli --role Role --command
     '{"Name" :  "pythonshell", "ScriptLocation" : "s3://amzn-s3-demo-bucket/
    scriptname.py"}'
         --connections Connections="connection-name" --default-arguments '{"--extra-
    py-files" : ["s3://amzn-s3-demo-bucket/EGG-FILE", "s3://amzn-s3-demo-bucket/WHEEL-
    FILE"]}'
    ```

    Wenn der Job ausgeführt wird, druckt das Skript die Zeilen, die in der *table_name* Tabelle im Amazon Redshift Redshift-Cluster erstellt wurden.

Verwendung AWS CloudFormation mit Python-Shell-Jobs in AWS Glue

Sie können AWS CloudFormation mit Python-Shell-Jobs in verwenden AWS Glue. Das Folgende ist ein Beispiel:

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Python39Job:
    Type: 'AWS::Glue::Job'
    Properties:
      Command:
        Name: pythonshell
        PythonVersion: '3.9'
        ScriptLocation: 's3://bucket/location'
      MaxRetries: 0
      Name: python-39-job
      Role: RoleName
```

Die Amazon CloudWatch Logs-Gruppe für die Ausgabe von Python-Shell-Jobs lautet`/aws-glue/python-jobs/output`. Informationen zu Fehlern finden Sie in der Protokollgruppe `/aws-glue/python-jobs/error`.

Migration von Python-Shell 3.6 auf Python-Shell 3.9

So migrieren Sie Ihre Python-Shell-Jobs auf die neueste AWS Glue Version:

1. Wählen Sie in der AWS Glue Konsole (https://console.aws.amazon.com/glue/) Ihren vorhandenen Python-Shell-Job aus.
2. Stellen Sie auf der Registerkarte Jobdetails die Python-Version auf ein `Python 3.9` und wählen Sie Speichern.
3. Stellen Sie sicher, dass Ihr Jobskript mit Python 3.9 kompatibel ist und erfolgreich ausgeführt wird.

Migrieren Sie von AWS Glue Python-Shell-Jobs

AWS hat 2018 die AWS Glue-Python-Shell-Jobs AWS gestartet und 2018 die AWS Glue-Python-Shell-Jobs gestartet, um Kunden eine einfache Möglichkeit zu bieten, Python-Skripte für small-to-medium große ETL-Jobs auszuführen und SQL-Abfragen auszulösen. Inzwischen gibt es jedoch modernere und flexiblere Optionen, um die Workloads zu bewältigen, mit denen derzeit gearbeitet PythonShell wird. In diesem Thema wird erklärt, wie Sie Ihre Workloads von AWS Glue Python Shell-Jobs zu einer dieser alternativen Optionen migrieren, um die neueren verfügbaren Funktionen zu nutzen.

In diesem Thema wird erklärt, wie Sie von AWS Glue-Python-Shell-Jobs zu alternativen Optionen migrieren.

Migration der Arbeitslast zu AWS Glue Spark-Jobs

AWS Glue Spark und PySpark Jobs ermöglichen es Ihnen, Ihre Workloads verteilt auszuführen. Da sowohl AWS Glue Python Shell-Jobs als auch AWS Glue Spark-Jobs auf derselben Plattform ausgeführt werden, ist die Migration einfach und Sie können weiterhin bestehende AWS Glue-Funktionen verwenden, die Sie mit Python Shell-Jobs verwenden, wie AWS Glue-Workflows, AWS Glue-Trigger, die Amazon EventBridge Integration von AWS Glue, die\ PIP-basierte Paketinstallation usw.

AWS Glue Spark-Jobs sind jedoch für die Ausführung von Spark-Workloads konzipiert, und die Mindestanzahl von Workern beträgt 2. Wenn Sie von Python-Shell-Jobs migrieren, ohne Ihre Skripte zu ändern, wird nur ein Worker tatsächlich verwendet, und die anderen Worker bleiben inaktiv. Dies wird Ihre Kosten erhöhen.

Um es effizient zu gestalten, schreiben Sie Ihr Python-Jobskript neu, um die Funktionen von Spark zu nutzen und die Arbeitslast auf mehrere Worker zu verteilen. Wenn Ihr Python-Skript auf Pandas basiert, ist die Migration mithilfe der New Pandas API auf Spark ganz einfach. Erfahren Sie mehr darüber im AWS Big Data-Blog: Tauchen Sie tief in AWS Glue 4.0 für Apache Spark ein.

Workload wird migriert zu AWS Lambda

AWS Lambda ist ein serverloser Computerdienst, mit dem Sie Code ausführen können, ohne Server bereitstellen oder verwalten zu müssen. Da AWS Lambda er kürzere Startzeiten und flexiblere Optionen für die Rechenkapazität bietet, können Sie von diesen Vorteilen profitieren. Für die Verwaltung zusätzlicher Python-Bibliotheken verwenden AWS Glue Python Shell-Jobs eine PIP-basierte Installation. Für müssen Sie AWS Lambda jedoch eine der folgenden Optionen wählen: ein ZIP-Archiv, ein Container-Image oder Lambda Layers.

Andererseits beträgt das maximale Timeout 900 Sekunden (15 Minuten). AWS Lambda Wenn die Jobdauer Ihres bestehenden AWS Glue Python Shell-Job-Workloads mehr als diese ist oder wenn Ihr Workload ein stacheliges Muster aufweist, das zu längeren Jobdauern führen kann, empfehlen wir, stattdessen andere Optionen zu prüfen. AWS Lambda

Migration von Workloads zu Amazon ECS/Fargate

Amazon Elastic Container Service (Amazon ECS) ist ein vollständig verwalteter Service, der die Bereitstellung, Verwaltung und Skalierung von containerisierten Anwendungen vereinfacht. AWS

Fargate ist eine serverlose Rechen-Engine für containerisierte Workloads, die auf Amazon ECS und Amazon Elastic Kubernetes Service (Amazon EKS) ausgeführt werden. Bei Amazon ECS und Fargate gibt es kein maximales Timeout, daher ist dies eine gute Option für Jobs mit langer Laufzeit. Da Sie die volle Kontrolle über Ihr Container-Image haben, können Sie Ihr Python-Skript und zusätzliche Python-Bibliotheken in den Container bringen und verwenden. Sie müssen Ihr Python-Skript jedoch containerisieren, um diesen Ansatz verwenden zu können.

Migration von Workloads zu Amazon Managed Workflows für Apache Airflow Python Operator

Amazon Managed Workflows for Apache Airflow (Managed Workflows for Apache Airflow) ist ein verwalteter Orchestrierungsservice für Apache Airflow, der die Einrichtung und den Betrieb von end-to-end Daten-Pipelines in der Cloud in großem Maßstab erleichtert. Wenn Sie bereits über eine MWAA-Umgebung verfügen, ist es einfach, den Python-Operator anstelle von AWS Glue-Python-Shell-Jobs zu verwenden. Der Python-Operator ist ein Operator, der Python-Code innerhalb eines Airflow-Workflows ausführt. Wenn Sie jedoch noch nicht über eine MWAA-Umgebung verfügen, empfehlen wir, andere Optionen zu prüfen.

Migration der Arbeitslast zu KI-Schulungsjobs Amazon SageMaker AI

Amazon SageMaker AI Training ist ein vollständig verwalteter Service für maschinelles Lernen (ML), der Ihnen hilft Amazon SageMaker AI , eine Vielzahl von ML-Modellen effizient und in großem Maßstab zu trainieren. Der Kern von Amazon SageMaker AI KI-Jobs ist die Containerisierung von ML-Workloads und die Fähigkeit, Rechenressourcen zu verwalten AWS . Wenn Sie eine serverlose Umgebung bevorzugen, in der es kein maximales Timeout gibt, könnten Amazon SageMaker AI KI-Schulungsjobs genau das Richtige für Sie sein. Die Startlatenz ist jedoch tendenziell länger als bei AWS Glue Python Shell-Jobs. Für Jobs, die latenzempfindlich sind, empfehlen wir, andere Optionen zu prüfen.

Überwachung AWS Glue

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer AWS Glue anderen AWS Lösungen. AWS bietet Überwachungstools, mit denen Sie beobachten AWS Glue, melden können, wenn etwas nicht stimmt, und bei Bedarf automatisch Maßnahmen ergreifen können:

Sie können die folgenden automatisierten Tools zur Überwachung von AWS Glue verwenden und möglicherweise auftretende Probleme melden:

- Amazon CloudWatch Events bietet einen Stream von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben, nahezu in Echtzeit. CloudWatch Events ermöglicht automatisiertes ereignisgesteuertes Rechnen. Sie können Regeln schreiben, die auf bestimmte Ereignisse achten und automatisierte Aktionen in anderen AWS Diensten auslösen, wenn diese Ereignisse eintreten. Weitere Informationen finden Sie im Amazon CloudWatch Events-Benutzerhandbuch.
- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien von EC2 Amazon-Instances und anderen Quellen überwachen AWS CloudTrail, speichern und darauf zugreifen. CloudWatch Logs kann Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.
- AWS CloudTrailerfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können feststellen, welche Benutzer und Konten anrufen AWS, von welcher Quell-IP-Adresse aus die Aufrufe getätigt werden und wann die Aufrufe erfolgen. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.

Darüber hinaus haben Sie in der AWS Glue Konsole Zugriff auf die folgenden Informationen, die Ihnen beim Debuggen und Profilieren von Jobs helfen:

- Spark-Jobs — Sie können eine Visualisierung ausgewählter CloudWatch Metrikreihen sehen, und neuere Jobs haben Zugriff auf die Spark-Benutzeroberfläche. Weitere Informationen finden Sie unter the section called “Überwachung von Spark-Aufträgen”.
- Ray-Jobs — Sie können eine Visualisierung ausgewählter CloudWatch Metrikreihen sehen. Weitere Informationen finden Sie unter the section called “Auftragsmetriken von Ray”.

Themen

- AWS Schlagworte in AWS Glue
- Automatisieren AWS Glue mit EventBridge
- AWS Glue Ressourcen überwachen
- Protokollierung AWS Glue API-Aufrufe mit AWS CloudTrail

AWS Schlagworte in AWS Glue

Um Ihnen bei der Verwaltung Ihrer zu helfen AWS Glue Sie können einigen Ressourcen optional Ihre eigenen Tags zuweisen AWS Glue Ressourcentypen. Ein Tag ist eine Bezeichnung, die Sie einer AWS Ressource zuweisen. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen. Sie können Tags verwenden in AWS Glue um Ihre Ressourcen zu organisieren und zu identifizieren. Tags können zur Erstellung von Kostenabrechnungsberichten und zur Einschränkung des Zugriffs auf Ressourcen verwendet werden. Wenn Sie verwenden AWS Identity and Access Management, können Sie steuern, welche Benutzer in Ihrem AWS Konto berechtigt sind, Tags zu erstellen, zu bearbeiten oder zu löschen. Zusätzlich zu den Berechtigungen zum Aufrufen von Tag-bezogenen Daten benötigen Sie auch die `glue:GetConnection` Berechtigung APIs, Tagging für Verbindungen APIs aufzurufen, und die `glue:GetDatabase` Berechtigung, Tagging APIs für Datenbanken aufzurufen. Weitere Informationen finden Sie unter ABAC mit Glue AWS.

In AWS Glue, können Sie die folgenden Ressourcen taggen:

- Verbindung
- Datenbank
- Crawler
- Interaktive Sitzung
- Entwicklungsendpunkt
- Aufgabe
- Auslöser
- Workflow
- Blueprint
- Machine-Learning-Transformation
- Regelsatz für die Datenqualität
- Stream-Schemas
- Stream-Schema-Registrierungen

> **Note**
>
> Als bewährte Methode, um diese mit Tags versehen zu können AWS Glue Ressourcen: Nehmen Sie die `glue:TagResource` Aktion immer in Ihre Richtlinien auf.

Beachten Sie Folgendes, wenn Sie Tags verwenden mit AWS Glue.

- Pro Entität werden maximal 50 Tags unterstützt.
- In AWS Glue, geben Sie Tags als eine Liste von Schlüssel-Wert-Paaren im folgenden Format an `{"string": "string" ...}`
- Wenn Sie ein Tag für ein Objekt erstellen, ist der Tag-Schlüssel erforderlich. Der Tag-Wert ist optional.
- Die Werte für Tag-Schlüssel und Tag-Wert unterscheiden zwischen Groß- und Kleinschreibung.
- Die Werte für Tag-Schlüssel und Tag-Wert dürfen nicht das Präfix awsenthalten. Für solche Tags sind keine Operationen zulässig.
- Die maximale Länge des Tag-Schlüssels beträgt 128 Unicode-Zeichen in UTF-8. Der Tag-Schlüssel darf nicht leer oder null sein.
- Die maximale Länge des Tag-Wertes beträgt 256 Unicode-Zeichen in UTF-8. Der Tag-Wert darf nicht leer oder null sein.

Tagging-Unterstützung für Verbindungen AWS Glue

Sie können `CreateConnection`-, `UpdateConnection`-, `GetConnection`- und `DeleteConnection`-Aktionsberechtigung basierend auf dem Ressourcen-Tag einschränken. Auf diese Weise können Sie die Zugriffskontrolle mit den geringsten Rechten für AWS Glue Jobs mit JDBC-Datenquellen implementieren, die JDBC-Verbindungsinformationen aus dem Datenkatalog abrufen müssen.

Beispielverwendung

Erstellen Sie eine AWS Glue Verbindung mit dem Tag ["connection-category“, „dev-test"].

Geben Sie die Tag-Bedingung für die `GetConnection`-Aktion in der IAM-Richtlinie an.

```
{
   "Effect": "Allow",
   "Action": [
```

```
      "glue:GetConnection"
      ],
    "Resource": "*",
    "Condition": {
      "ForAnyValue:StringEquals": {
        "aws:ResourceTag/tagKey": "dev-test"
      }
    }
  }
```

Beispiele

Die folgenden Beispiele erstellen eine Aufgabe mit zugewiesenen Tags.

AWS CLI

```
aws glue create-job --name job-test-tags --role MyJobRole --command
 Name=glueetl,ScriptLocation=S3://aws-glue-scripts//prod-job1
--tags key1=value1,key2=value2
```

AWS CloudFormation JSON

```
{
  "Description": "AWS Glue Job Test Tags",
  "Resources": {
    "MyJobRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "glue.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
```

```
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "root",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": "*",
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    },
    "MyJob": {
      "Type": "AWS::Glue::Job",
      "Properties": {
        "Command": {
          "Name": "glueetl",
          "ScriptLocation": "s3://aws-glue-scripts//prod-job1"
        },
        "DefaultArguments": {
          "--job-bookmark-option": "job-bookmark-enable"
        },
        "ExecutionProperty": {
          "MaxConcurrentRuns": 2
        },
        "MaxRetries": 0,
        "Name": "cf-job1",
        "Role": {
           "Ref": "MyJobRole",
           "Tags": {
                "key1": "value1",
                "key2": "value2"
           }
        }
      }
    }
  }
```

```
}
```

AWS CloudFormation YAML

```
Description: AWS Glue Job Test Tags
Resources:
  MyJobRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - glue.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: "/"
      Policies:
        - PolicyName: root
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
                Action: "*"
                Resource: "*"
  MyJob:
    Type: AWS::Glue::Job
    Properties:
      Command:
        Name: glueetl
        ScriptLocation: s3://aws-glue-scripts//prod-job1
      DefaultArguments:
        "--job-bookmark-option": job-bookmark-enable
      ExecutionProperty:
        MaxConcurrentRuns: 2
      MaxRetries: 0
      Name: cf-job1
      Role:
        Ref: MyJobRole
        Tags:
           key1: value1
```

```
            key2: value2
```

Weitere Informationen finden Sie unter Tagging-Strategien in AWS.

Informationen dazu, wie Sie den Zugriff mithilfe von Tags steuern, finden Sie unter ABAC mit Glue AWS.

Automatisieren AWS Glue mit EventBridge

Sie können Amazon verwenden EventBridge , um Ihre AWS Services zu automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen zu reagieren. Ereignisse aus AWS Services werden nahezu EventBridge in Echtzeit übermittelt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen durchgeführt werden sollen, wenn sich für ein Ereignis eine Übereinstimmung mit einer Regel ergibt. Die folgenden Aktionen können beispielsweise automatisch ausgelöst werden:

- Eine AWS Lambda Funktion aufrufen
- Amazon EC2 Run Command aufrufen
- Weiterleiten des Ereignisses an Amazon Kinesis Data Streams
- Aktivierung einer AWS Step Functions Zustandsmaschine
- Benachrichtigen eines Amazon SNS-Themas oder einer Amazon SQS-Warteschlange

Einige Beispiele für die Verwendung EventBridge mit AWS Glue sind Folgende:

- Aktivieren einer Lambda-Funktion, wenn ein ETL-Auftrag erfolgreich ist
- Benachrichtigen eines Amazon-SNS-Themas, wenn ein ETL-Auftrag fehlschlägt

Folgendes EventBridge wird generiert von AWS Glue.

- Ereignisse für `"detail-type":"Glue Job State Change"` werden bei `SUCCEEDED`, `FAILED`, `TIMEOUT` und `STOPPED` generiert.
- Ereignisse für `"detail-type":"Glue Job Run Status"` werden generiert für `RUNNING`, `STARTING` und `STOPPING` Auftragsläufe, wenn sie die Benachrichtigungsschwelle für die Auftragsverzögerung überschreiten. Sie müssen die Schwellenwerteigenschaft für die Benachrichtigung über einen Auftrag festlegen, um diese Ereignisse zu empfangen.

Es wird nur ein Ereignis pro Status der Auftragsausführung generiert, wenn die Schwelle für die Benachrichtigung über die Auftragsverzögerung überschritten wird.

- Ereignisse für `"detail-type":"Glue Crawler State Change"` werden bei `Started`, `Succeeded` und `Failed` generiert.
- Ereignisse für `“detail_type”:“Glue Scheduled Crawler Invocation Failure”` werden generiert, wenn der geplante Crawler nicht gestartet werden kann. In den Details der Benachrichtigung:
  - `customerId`enthält die Konto-ID des Kunden.
  - `crawlerName`enthält den Namen des Crawlers, der nicht gestartet werden konnte.
  - `errorMessage`enthält die Ausnahmemeldung des fehlgeschlagenen Aufrufs.
- Ereignisse für `“detail_type”:“Glue Auto Statistics Invocation Failure“` werden generiert, wenn die Ausführung der Aufgabe für automatisch verwaltete Spaltenstatistiken nicht gestartet werden kann. In den Details der Benachrichtigung:
  - `catalogId`enthält die einem Katalog zugeordnete ID.
  - `databaseName`enthält den Namen der betroffenen Datenbank.
  - `tableName`enthält den Namen der betroffenen Tabelle.
  - `errorMessage`enthält die Ausnahmemeldung des fehlgeschlagenen Aufrufs.
- Ereignisse für `“detail_type”:“Glue Scheduled Statistics Invocation Failure”` werden generiert, wenn die Ausführung der geplanten (Cron-) Aufgabe für Spaltenstatistiken nicht gestartet werden kann. In den Details der Benachrichtigung:
  - `catalogId`enthält die einem Katalog zugeordnete ID.
  - `databaseName`enthält den Namen der betroffenen Datenbank.
  - `tableName`enthält den Namen der betroffenen Tabelle.
  - `errorMessage`enthält die Ausnahmemeldung des fehlgeschlagenen Aufrufs.
- Ereignisse für `“detail_type”:“Glue Statistics Task Started”` werden generiert, wenn die Ausführung der Aufgabe für Spaltenstatistiken gestartet wird.
- Ereignisse für `“detail_type”:“Glue Statistics Task Succeeded”` werden generiert, wenn die Ausführung der Aufgabe für Spaltenstatistiken erfolgreich ist.
- Ereignisse für `“detail_type”:“Glue Statistics Task Failed”` werden generiert, wenn die Ausführung der Aufgabe für Spaltenstatistiken fehlschlägt.
- Ereignisse für `"detail-type":"Glue Data Catalog Database State Change"` werden für `CreateDatabase`, `DeleteDatabase`, `CreateTable`, `DeleteTable` und

`BatchDeleteTable` generiert. Wenn beispielsweise eine Tabelle erstellt oder gelöscht wird, wird eine Benachrichtigung an gesendet EventBridge. Beachten Sie, dass Sie kein Programm schreiben können, das von der Reihenfolge oder dem Vorhandensein von Benachrichtigungsereignissen abhängig ist, da Benachrichtigungen möglicherweise nicht der Reihe nach erfolgen oder fehlen. Ereignisse werden auf bestmögliche Weise ausgegeben. In den Details der Benachrichtigung:

- Der Abschnitt `typeOfChange` enthält den Namen der API-Operation.
- Der Abschnitt `databaseName` enthält den Namen der betroffenen Datenbank.
- Der Abschnitt `changedTables` enthält bis zu 100 Namen von betroffenen Tabellen pro Benachrichtigung. Wenn Tabellennamen zu lang sind, werden möglicherweise mehrere Benachrichtigungen erstellt.

- Ereignisse für `"detail-type":"Glue Data Catalog Table State Change"` werden für `UpdateTable`, `CreatePartition`, `BatchCreatePartition`, `UpdatePartition`, `DeletePartition`, `BatchUpdatePartition` und `BatchDeletePartition`. generiert. Wenn beispielsweise eine Tabelle oder Partition aktualisiert wird, wird eine Benachrichtigung an gesendet EventBridge. Beachten Sie, dass Sie kein Programm schreiben können, das von der Reihenfolge oder dem Vorhandensein von Benachrichtigungsereignissen abhängig ist, da Benachrichtigungen möglicherweise nicht der Reihe nach erfolgen oder fehlen. Ereignisse werden auf bestmögliche Weise ausgegeben. In den Details der Benachrichtigung:
  - Der Abschnitt `typeOfChange` enthält den Namen der API-Operation.
  - Der Abschnitt `databaseName` enthält den Namen der Datenbank, die die betroffenen Ressourcen enthält.
  - Der Abschnitt `tableName` enthält den Namen der betroffenen Tabelle.
  - Der Abschnitt `changedPartitions` gibt bis zu 100 betroffene Partitionen in einer einzigen Benachrichtigung an. Wenn Partitionsnamen zu lang sind, werden möglicherweise mehrere Benachrichtigungen erstellt.

    Wenn es beispielsweise zwei Partitionsschlüssel gibt, `Year` und `Month`, ändert `"2018,01", "2018,02"` die Partition mit `"Year=2018" and "Month=01"` und die Partition mit `"Year=2018" and "Month=02"`.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Glue Data Catalog Table State Change",
    "source":"aws.glue",
    "account":"123456789012",
```

```
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":["arn:aws:glue:us-west-2:123456789012:database/default/foo"],
    "detail":{
        "changedPartitions": [
            "2018,01",
            "2018,02"
        ],
        "databaseName": "default",
        "tableName": "foo",
        "typeOfChange": "BatchCreatePartition"
        }
}
```

Weitere Informationen finden Sie im Amazon CloudWatch Events-Benutzerhandbuch. Für spezifische Veranstaltungen für AWS Glue, siehe AWS Glue Ereignisse.

AWS Glue Ressourcen überwachen

AWS Glue verfügt über Servicebeschränkungen, um Kunden vor unerwarteter exzessiver Bereitstellung und vor böswilligen Aktionen zu schützen, die Ihre Rechnung in die Höhe treiben sollen. Die Limits schützen auch den Service. Wenn sich Kunden an der AWS Service Quota-Konsole anmelden, können sie ihre aktuellen Ressourcenlimits einsehen und gegebenenfalls eine Erhöhung beantragen.

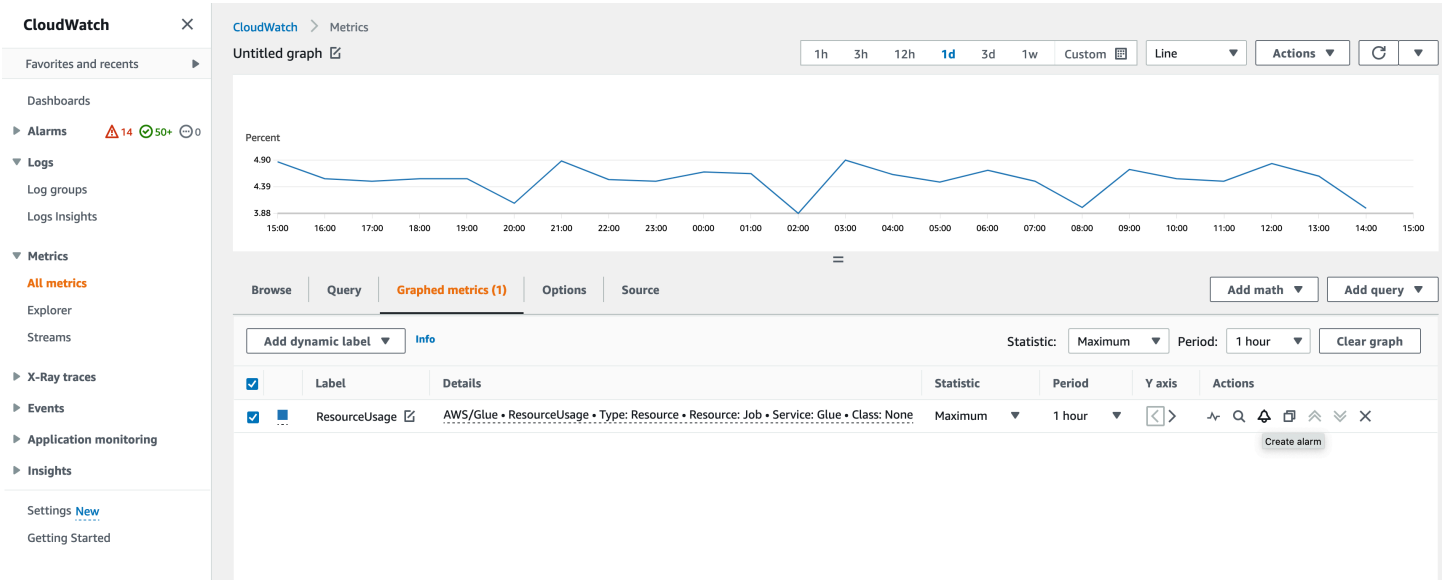
AWS Glue ermöglicht es Ihnen, den Ressourcenverbrauch des Services in Amazon als Prozentsatz anzuzeigen CloudWatch und CloudWatch Alarme zur Überwachung der Nutzung zu konfigurieren. Amazon CloudWatch bietet die Überwachung der AWS Ressourcen und der Kundenanwendungen, die auf der Amazon-Infrastruktur ausgeführt werden. Die Metriken sind für Sie kostenlos. Die folgenden Metriken werden unterstützt:

- Anzahl der Workflows pro Konto
- Anzahl der Auslöser pro Konto
- Anzahl der Aufgaben pro Konto
- Anzahl der gleichzeitigen Aufgabenausführungen pro Konto
- Anzahl der Vorlagen pro Konto
- Anzahl der interaktiven Sitzungen pro Konto

Konfiguration und Verwendung von Ressourcenmetriken

Um diese Funktion zu nutzen, können Sie zur CloudWatch Amazon-Konsole gehen, um die Metriken anzusehen und Alarme zu konfigurieren. Die Metriken befinden sich im AWS/Glue-Namespace und stellen einen Prozentsatz der tatsächlichen Ressourcennutzung geteilt durch das Ressourcenkontingent dar. Die CloudWatch Messwerte werden an Ihre Konten übermittelt, was für Sie kostenlos ist. Wenn Sie beispielsweise 10 Workflows erstellt haben und Ihr Servicekontingent maximal 200 Workflows zulässt, beträgt Ihre Nutzung 10/200 = 5 %, und im Diagramm sehen Sie einen Datenpunkt von 5 als Prozentsatz. Um genauer zu sein:

```
Namespace: AWS/Glue
Metric name: ResourceUsage
Type: Resource
Resource: Workflow (or Trigger, Job, JobRun, Blueprint, InteractiveSession)
Service: Glue
Class: None
```



Um einen Alarm für eine Metrik in der CloudWatch Konsole zu erstellen:

1. Sobald Sie die Metrik gefunden haben, gehen Sie zu Grafische Metriken.
2. Klicken Sie auf Alarm erstellen unter Aktionen.
3. Konfigurieren Sie den Alarm nach Bedarf.

Wir geben Metriken aus, wenn sich Ihre Ressourcennutzung ändert (z. B. eine Zunahme oder Abnahme). Wenn sich Ihre Ressourcennutzung jedoch nicht ändert, geben wir stündlich Messwerte

aus, sodass Sie ein kontinuierliches CloudWatch Diagramm erhalten. Um fehlende Datenpunkte zu vermeiden, empfehlen wir Ihnen nicht, einen Zeitraum von weniger als 1 Stunde zu konfigurieren.

Sie können Alarme auch AWS CloudFormation wie im folgenden Beispiel konfigurieren. In diesem Beispiel wird, sobald die Workflow-Ressourcennutzung 80 % erreicht, ein Alarm ausgelöst, um eine Nachricht an das vorhandene SNS-Thema zu senden, wo Sie es abonnieren können, um Benachrichtigungen zu erhalten.

```
{
 "Type": "AWS::CloudWatch::Alarm",
 "Properties": {
  "AlarmName": "WorkflowUsageAlarm",
  "ActionsEnabled": true,
  "OKActions": [],
  "AlarmActions": [
   "arn:aws:sns:af-south-1:085425700061:Default_CloudWatch_Alarms_Topic"
  ],
  "InsufficientDataActions": [],
  "MetricName": "ResourceUsage",
  "Namespace": "AWS/Glue",
  "Statistic": "Maximum",
  "Dimensions": [{
    "Name": "Type",
    "Value": "Resource"
   },
   {
    "Name": "Resource",
    "Value": "Workflow"
   },
   {
    "Name": "Service",
    "Value": "Glue"
   },
   {
    "Name": "Class",
    "Value": "None"
   }
  ],
  "Period": 3600,
  "EvaluationPeriods": 1,
  "DatapointsToAlarm": 1,
  "Threshold": 80,
  "ComparisonOperator": "GreaterThanThreshold",
```

```
  "TreatMissingData": "notBreaching"
 }
}
```

Protokollierung AWS Glue API-Aufrufe mit AWS CloudTrail

AWS Glue ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Dienst ausgeführt wurden AWS Glue. CloudTrail erfasst alle API-Aufrufe für AWS Glue als Ereignisse. Zu den erfassten Anrufen gehören Anrufe von AWS Glue Konsolen- und Codeanrufe an AWS Glue API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für AWS Glue. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf einsehen. Anhand der von CloudTrail gesammelten Informationen können Sie ermitteln, welche Anfrage gestellt wurde AWS Glue, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

AWS Glue Informationen in CloudTrail

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn Aktivität stattfindet in AWS Glue, diese Aktivität wird zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto ansehen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS Konto finden Sie, einschließlich Ereignisse für AWS Glue, erstelle eine Spur. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole erstellen, gilt der Trail standardmäßig für alle AWS Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Einen Trail für dein AWS Konto erstellen
- CloudTrail unterstützte Dienste und Integrationen

- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

Alle AWS Glue Aktionen werden von protokolliert CloudTrail und sind in der dokumentiertAWS Glue API. Beispielsweise generieren Aufrufe von `CreateTable` und `CreateScript` Aktionen Einträge in den CloudTrail Protokolldateien. `CreateDatabase`

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anforderung mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Protokolliert jedoch CloudTrail nicht alle Informationen zu Anrufen. Beispielsweise werden bestimmte vertrauliche Informationen nicht protokolliert, z. B. die, die in Verbindungsanfragen `ConnectionProperties` verwendet werden, und es protokolliert a `null` anstelle der Antworten, die von den folgenden Antworten zurückgegeben werden APIs:

```
BatchGetPartition       GetCrawlers          GetJobs          GetTable
CreateScript            GetCrawlerMetrics    GetJobRun        GetTables
GetCatalogImportStatus  GetDatabase          GetJobRuns       GetTableVersions
GetClassifier           GetDatabases         GetMapping       GetTrigger
GetClassifiers          GetDataflowGraph     GetObjects       GetTriggers
GetConnection           GetDevEndpoint       GetPartition     GetUserDefinedFunction
GetConnections          GetDevEndpoints      GetPartitions    GetUserDefinedFunctions
GetCrawler              GetJob               GetPlan
```

Verstehen AWS Glue Einträge in Protokolldateien

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion,

Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `DeleteCrawler` Aktion demonstriert.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "johndoe"
  },
  "eventTime": "2017-10-11T22:29:49Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "DeleteCrawler",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "72.21.198.64",
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "name": "tes-alpha"
  },
  "responseElements": null,
  "requestID": "b16f4050-aed3-11e7-b0b3-75564a46954f",
  "eventID": "e73dd117-cfd1-47d1-9e2f-d1271cad838c",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

Dieses Beispiel zeigt einen CloudTrail Protokolleintrag, der eine `CreateConnection` Aktion demonstriert.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "userName": "johndoe"
  },
  "eventTime": "2017-10-13T00:19:19Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "CreateConnection",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "72.21.198.66",
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "connectionInput": {
      "name": "test-connection-alpha",
      "connectionType": "JDBC",
      "physicalConnectionRequirements": {
        "subnetId": "subnet-323232",
        "availabilityZone": "us-east-1a",
        "securityGroupIdList": [
          "sg-12121212"
        ]
      }
    }
  },
  "responseElements": null,
  "requestID": "27136ebc-afac-11e7-a7d6-ab217e5c3f19",
  "eventID": "e8b3baeb-c511-4597-880f-c16210c60a4a",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

AWS Glue Streamen

AWS Glue Streaming, eine Komponente von AWS Glue, ermöglicht es Ihnen, Streaming-Daten effizient und nahezu in Echtzeit zu verarbeiten, sodass Sie wichtige Aufgaben wie Datenaufnahme, -verarbeitung und maschinelles Lernen ausführen können. Mithilfe des Apache Spark Streaming-Frameworks bietet AWS Glue Streaming einen serverlosen Dienst, der Streaming-Daten in großem Umfang verarbeiten kann. AWS Glue bietet zusätzlich zu Apache Spark verschiedene Optimierungen wie serverlose Infrastruktur, auto-scaling, visuelle Jobentwicklung, Instant-On-Notebooks für Streaming-Jobs und andere Leistungsverbesserungen.

Anwendungsfälle für Streaming

Zu den häufigsten Anwendungsfällen für Streaming gehören: AWS Glue

Near-real-time Datenverarbeitung: Mit AWS Glue Streaming können Unternehmen Streaming-Daten nahezu in Echtzeit verarbeiten, sodass sie Erkenntnisse gewinnen und zeitnahe Entscheidungen auf der Grundlage der neuesten Informationen treffen können.

Betrugserkennung: Sie können AWS Glue Streaming für die Echtzeitanalyse von Streaming-Daten nutzen, was es für die Erkennung betrügerischer Aktivitäten wie Kreditkartenbetrug, Netzwerkeinbrüche oder Online-Betrug wertvoll macht. Durch die kontinuierliche Verarbeitung und Analyse der eingehenden Daten können Sie verdächtige Muster oder Anomalien schnell erkennen.

Analyse sozialer Medien: Beim AWS Glue Streaming können Social-Media-Daten wie Tweets, Beiträge oder Kommentare in Echtzeit verarbeitet werden, sodass Unternehmen Trends beobachten, Stimmungsanalysen durchführen und den Ruf der Marke in Echtzeit verwalten können.

Internet of Things (IoT) -Analytik: AWS Glue Streaming eignet sich für die Verarbeitung und Analyse von Hochgeschwindigkeitsdatenströmen, die von IoT-Geräten, Sensoren und angeschlossenen Maschinen generiert werden. Es ermöglicht Echtzeit-Überwachung, Anomalie-Erkennung, prädiktive Wartung und andere IoT-Analytik-Anwendungen.

Clickstream-Analyse: AWS Glue Streaming kann Clickstream-Daten von Websites oder mobilen Anwendungen in Echtzeit verarbeiten und analysieren. Dies ermöglicht es Unternehmen, Einblicke in das Benutzerverhalten zu gewinnen, das Benutzererlebnis zu personalisieren und Marketingkampagnen auf der Grundlage von Clickstream-Daten in Echtzeit zu optimieren.

Protokollüberwachung und -analyse: AWS Glue Streaming kann Protokolldaten von Servern, Anwendungen oder Netzwerkgeräten kontinuierlich in Echtzeit verarbeiten und analysieren. Dies

hilft bei der Erkennung von Anomalien, der Behebung von Problemen und der Überwachung von Systemzustand und Leistung.

Empfehlungssysteme: AWS Glue Streaming kann Benutzeraktivitätsdaten in Echtzeit verarbeiten und Empfehlungsmodelle dynamisch aktualisieren. Dies ermöglicht personalisierte Empfehlungen in Echtzeit, die auf dem Verhalten und den Vorlieben der Benutzer basieren.

Dies sind einige Beispiele für die vielfältigen Anwendungsfälle, in denen AWS Glue Streaming angewendet werden kann. Die Integration in das AWS Ökosystem und die verwalteten Dienste machen es zu einer bequemen Wahl für die Verarbeitung und Analyse von Streams in Echtzeit in der Cloud.

Was sind die Vorteile der Nutzung von AWS Glue Streaming?

Die Nutzung von AWS Glue Streaming bietet folgende Vorteile:

- Serverlos: AWS Glue Streaming ist serverlos, sodass die Infrastruktur nicht verwaltet werden muss. Dadurch wird der betriebliche Aufwand reduziert und Benutzer können sich auf die Datenverarbeitung und Analyseaufgaben konzentrieren, anstatt die Infrastruktur verwalten zu müssen.
- Autoscaling: AWS Glue Streaming bietet Autoscaling-Funktionen, bei denen die Verarbeitungskapazität dynamisch an die Arbeitslast angepasst wird. Es wird automatisch auf- oder abskalieren, um Schwankungen im Datenvolumen auszugleichen und eine optimale Leistung und Ressourcennutzung zu gewährleisten.
- Visuelle Entwicklung: Die Entwicklung von Streaming-Jobs kann komplex sein. AWS Glue Streaming begegnet dieser Herausforderung durch das Angebot von AWS Glue Studio, einem visuellen Authoring-Tool. AWS Glue Studio vereinfacht den Prozess der Erstellung von Streaming-Workflows und ermöglicht es Entwicklern, Streaming-Anwendungen visuell zu entwerfen und zu verwalten, wodurch die Lernkurve reduziert und die Produktivität gesteigert wird.
- Kostengünstig: Als serverloser Dienst bietet AWS Glue Streaming Kosteneffizienz, da die Bereitstellung und Wartung der Infrastruktur überflüssig wird. Die Abrechnung erfolgt auf der Grundlage der bei der Ausführung von Streaming-Aufträgen verbrauchten Ressourcen. Dies ermöglicht eine Kostenoptimierung und Skalierung anhand der tatsächlichen Nutzung.
- Bewältigt komplexe Workloads: AWS Glue Streaming ist darauf ausgelegt, komplexe Streaming-Workloads zu bewältigen. Es kann große Mengen an Echtzeitdaten verarbeiten und analysieren, erweiterte Transformationen unterstützen und in andere AWS Dienste integrieren, wodurch ausgeklügelte Streaming-Daten-Pipelines und Analyse-Workflows ermöglicht werden.

- Keine Bindung: AWS Glue Streaming bietet Flexibilität und vermeidet die Bindung an einen Anbieter. Benutzer können AWS Glue Streaming als Teil des breiteren AWS Ökosystems nutzen und es nahtlos in andere AWS Dienste integrieren. Dies ermöglicht eine einfache Integration mit bestehenden Datenquellen, Anwendungen und Services, ohne an eine bestimmte Technologie oder Plattform gebunden zu sein.

Wann sollte AWS Glue Streaming verwendet werden?

Es gibt viele Optionen, wenn es um Streaming-Anwendungsfälle geht. Wir empfehlen AWS Glue Streaming in den folgenden Szenarien.

1. Wenn Sie Spark bereits für die Stapelverarbeitung verwenden AWS Glue , ist AWS Glue Streaming die ideale Wahl für Sie. Es bietet einen nahtlosen Übergang zur Erstellung von Streaming-Aufträgen, ohne dass Sie eine neue Sprache oder ein neues Framework lernen müssen. AWS Glue Streaming nutzt Ihr vorhandenes Wissen und Ihre Infrastruktur, vereinfacht den Prozess der Jobentwicklung und ermöglicht es Ihnen, Ihre Datenverarbeitungskapazitäten auf einfache Weise auf Echtzeit-Streaming-Szenarien auszudehnen.
2. Wenn Sie einen einheitlichen Service oder ein einheitliches Produkt für die Verarbeitung von Batch-, Streaming- und ereignisgesteuerten Workloads benötigen, ist AWS Glue Streaming die richtige Lösung für Sie. Mit AWS Glue Streaming können Sie Ihre Datenverarbeitungsanforderungen in einem einzigen Framework konsolidieren, wodurch die Komplexität der Verwaltung mehrerer Systeme entfällt. Dies ermöglicht eine effiziente Entwicklung und Pflege verschiedener Daten-Workflows und gewährleistet gleichzeitig Konsistenz und Kompatibilität über verschiedene Workload-Typen hinweg.
3. AWS Glue Streaming eignet sich gut für Szenarien mit extrem großen Streaming-Datenmengen und komplexen Transformationen, wie Verknüpfungen zwischen Streams oder relationalen Datenbanken. Es kann riesige Datenströme effizient verarbeiten und analysieren, so dass Sie auch anspruchsvolle Workloads mühelos bewältigen können. Ganz gleich, ob es sich um eine schnelle Datenaufnahme oder komplizierte Datenmanipulationen handelt, die Skalierbarkeit und die fortschrittlichen Verarbeitungsfunktionen von AWS Glue Streaming sorgen für optimale Leistung und genaue Ergebnisse.
4. Wenn Sie bei der Erstellung von Streaming-Jobs einen visuellen Ansatz bevorzugen, AWS Glue bietet AWS Glue Studio, mit dem Sie Ihre Streaming-Anwendungen visuell entwerfen und verwalten können, wodurch der Entwicklungsprozess vereinfacht wird. Diese intuitive Oberfläche ermöglicht es Entwicklern, Streaming-Workflows über eine visuelle Oberfläche zu erstellen, zu

konfigurieren und zu überwachen, wodurch die Lernkurve gesenkt und die Produktivität erhöht wird.

5. AWS Glue Streaming ist eine hervorragende Wahl für near-real-time Anwendungsfälle, in denen verbindliche Vereinbarungen SLAs (Service Level Agreements) von mehr als 10 Sekunden gelten.
6. Wenn Sie einen transaktionalen Data Lake mit Apache Iceberg, Apache Hudi oder Delta Lake erstellen, bietet AWS Glue Streaming native Unterstützung für diese offenen Tabellenformate. Diese nahtlose Integration ermöglicht es Ihnen, Streaming-Daten direkt aus diesen transaktionalen Data Lakes zu verarbeiten und so die Datenkonsistenz, -integrität und -kompatibilität sicherzustellen.
7. Wenn Streaming-Daten für eine Vielzahl von Datenzielen aufgenommen werden müssen: AWS Glue Streaming bietet native Ziele für eine Vielzahl von Datenzielen wie Amazon Redshift, Amazon RDS, Amazon Aurora, Oracle, SQL Server und andere Ziele.

Unterstützte Datenquellen

AWS Glue Streaming unterstützt die folgenden Datenquellen:

- Amazon Kinesis
- Amazon MSK (Managed Streaming für Apache Kafka)
- Selbstverwaltetes Apache Kafka

Unterstützte Datenziele

AWS Glue Streaming unterstützt eine Vielzahl von Datenzielen wie:

- Von Data Catalog unterstützte AWS Glue Datenziele
- Amazon S3
- Amazon Redshift
- MySQL
- PostgreSQL
- Oracle
- Microsoft SQL Server
- Snowflake

- Jede Datenbank, die mit JDBC verbunden werden kann
- Apache Iceberg, Delta und Apache Hudi
- AWS Glue Marketplace-Konnektoren

Tutorial: Erstellen Sie Ihren ersten Streaming-Workload mit AWS Glue Studio

In diesem Tutorial erfahren Sie, wie Sie mit AWS Glue Studio einen Streaming-Job erstellen. AWS Glue Studio ist eine visuelle Oberfläche zum Erstellen von AWS Glue Jobs.

Sie können Streaming-Aufträge zu Extract, Transform, Load (ETL) erstellen, die kontinuierlich ausgeführt werden und Daten aus Streaming-Quellen in Amazon Kinesis Data Streams, Apache Kafka und Amazon Managed Streaming für Apache Kafka (Amazon MSK) nutzen.

Voraussetzungen

Um diesem Tutorial zu folgen, benötigen Sie einen Benutzer mit AWS Konsolenberechtigungen zur Nutzung AWS Glue von Amazon Kinesis, Amazon S3, Amazon Athena AWS CloudFormation, AWS Lambda und Amazon Cognito.

Streaming-Daten von Amazon Kinesis verarbeiten

Themen

- Generieren von Scheindaten mit Kinesis Data Generator
- Einen AWS Glue Streaming-Job mit AWS Glue Studio erstellen
- Durchführen einer Transformation und Speichern des transformierten Ergebnisses in Amazon S3

Generieren von Scheindaten mit Kinesis Data Generator

Mit dem Kinesis Data Generator (KDG) können Sie Beispieldaten im JSON-Format synthetisch erzeugen. Vollständige Anweisungen und Details finden Sie in der Dokumentation des Tools.

1. Um zu beginnen, klicken Sie hier,

   um eine AWS CloudFormation Vorlage in Ihrer Umgebung auszuführen. AWS

> **Note**
>
> Möglicherweise tritt ein CloudFormation Vorlagenfehler auf, weil einige Ressourcen, wie z. B. der Amazon Cognito Cognito-Benutzer für Kinesis Data Generator, bereits in Ihrem AWS Konto vorhanden sind. Das könnte daran liegen, dass Sie dies bereits in einem anderen Tutorial oder Blog eingerichtet haben. Um dieses Problem zu beheben, können Sie entweder die Vorlage für einen Neuanfang in einem neuen AWS Konto ausprobieren oder eine andere AWS Region erkunden. Mit diesen Optionen können Sie das Tutorial ausführen, ohne mit bestehenden Ressourcen in Konflikt zu geraten.

Die Vorlage stellt einen Kinesis-Datenstrom und ein Kinesis-Data-Generator-Konto für Sie bereit. Es erstellt außerdem einen Amazon-S3-Bucket für die Daten und eine Glue-Servicerolle mit der für dieses Tutorial erforderlichen Berechtigung.

2. Geben Sie einen Benutzernamen und ein Passwort ein, mit denen sich der KDG authentifizieren soll. Notieren Sie sich den Benutzernamen und das Passwort für die weitere Verwendung.
3. Wählen Sie Weiter bis zum letzten Schritt. Bestätigen Sie die Erstellung von IAM-Ressourcen. Suchen Sie oben auf dem Bildschirm nach Fehlern, z. B. wenn das Passwort nicht den Mindestanforderungen entspricht, und stellen Sie die Vorlage bereit.
4. Navigieren Sie zur Registerkarte Ausgaben des Stacks. Sobald die Vorlage bereitgestellt ist, wird die generierte Eigenschaft angezeigt KinesisDataGeneratorUrl. Klicken Sie auf diese URL.
5. Geben Sie den Benutzernamen und das Passwort ein, die Sie sich notiert haben.
6. Wählen Sie die Region aus, die Sie verwenden, und wählen Sie den Kinesis-Stream `GlueStreamTest-{AWS::AccountId}`
7. Geben Sie die folgende Vorlage ein:

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
            "min":5,
            "max":30
        }
    )}},
```

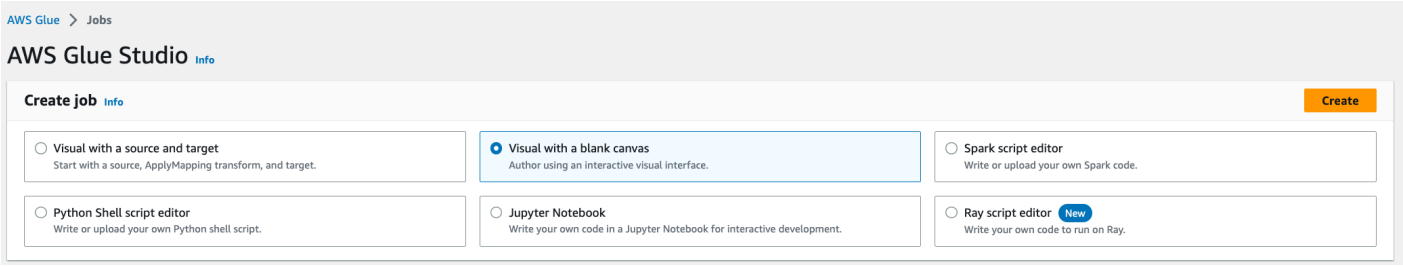
```
    "o2stats": {{random.number(
        {
            "min":92,
            "max":98
        }
    )}},
    "minutevolume": {{random.number(
        {
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Sie können jetzt Scheindaten mit der Testvorlage anzeigen und die Scheindaten mit Daten senden in Kinesis aufnehmen.

8. Klicken Sie auf Daten senden und generieren Sie 5–10 000 Datensätze für Kinesis.

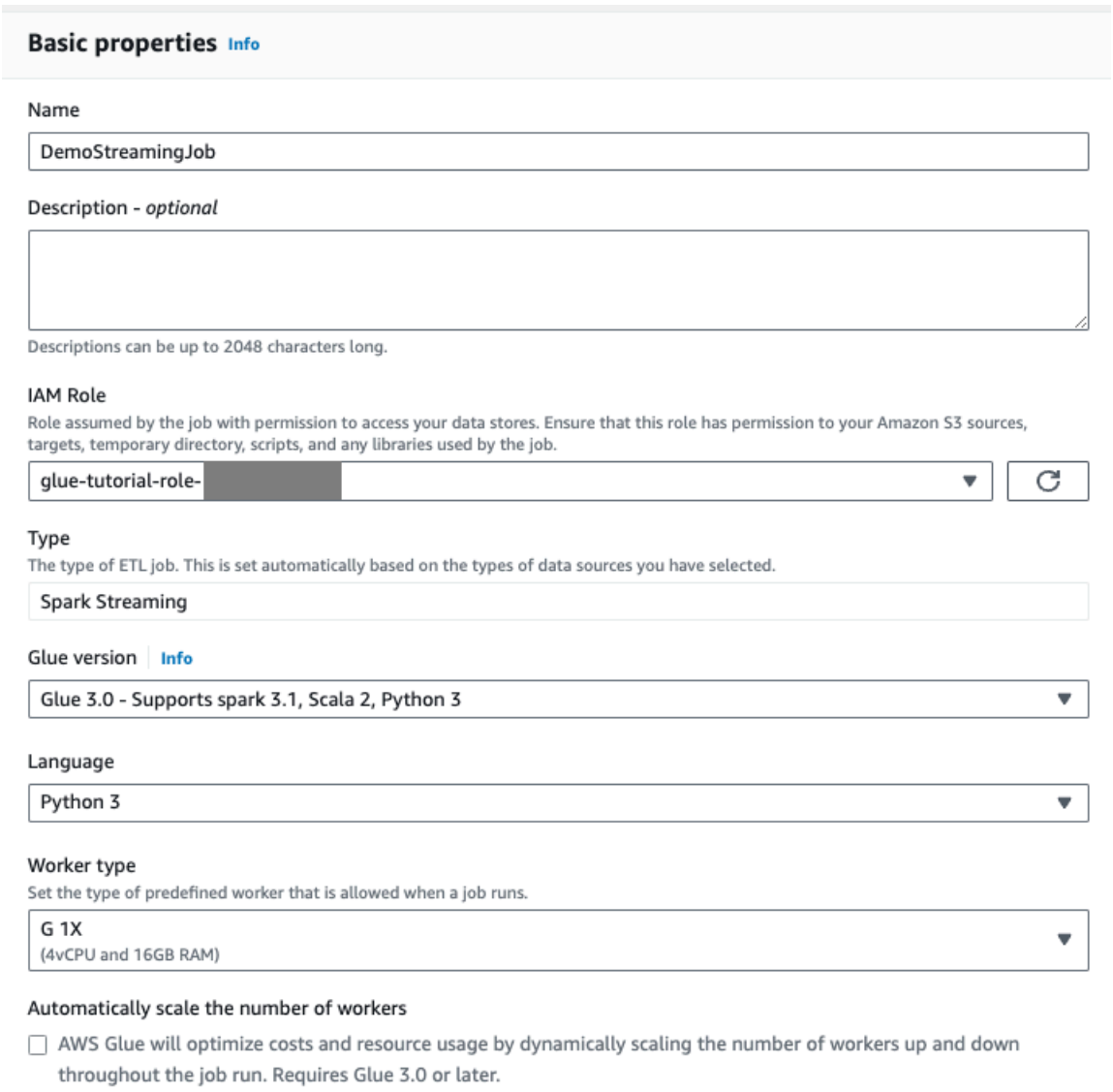
Einen AWS Glue Streaming-Job mit AWS Glue Studio erstellen

1. Navigieren Sie AWS Glue in der Konsole zu derselben Region.
2. Wählen Sie in der linken Navigationsleiste unter Datenintegration und ETL die Option ETL-Aufträge.
3. Erstellen Sie einen AWS Glue Job über Visual mit einer leeren Leinwand.

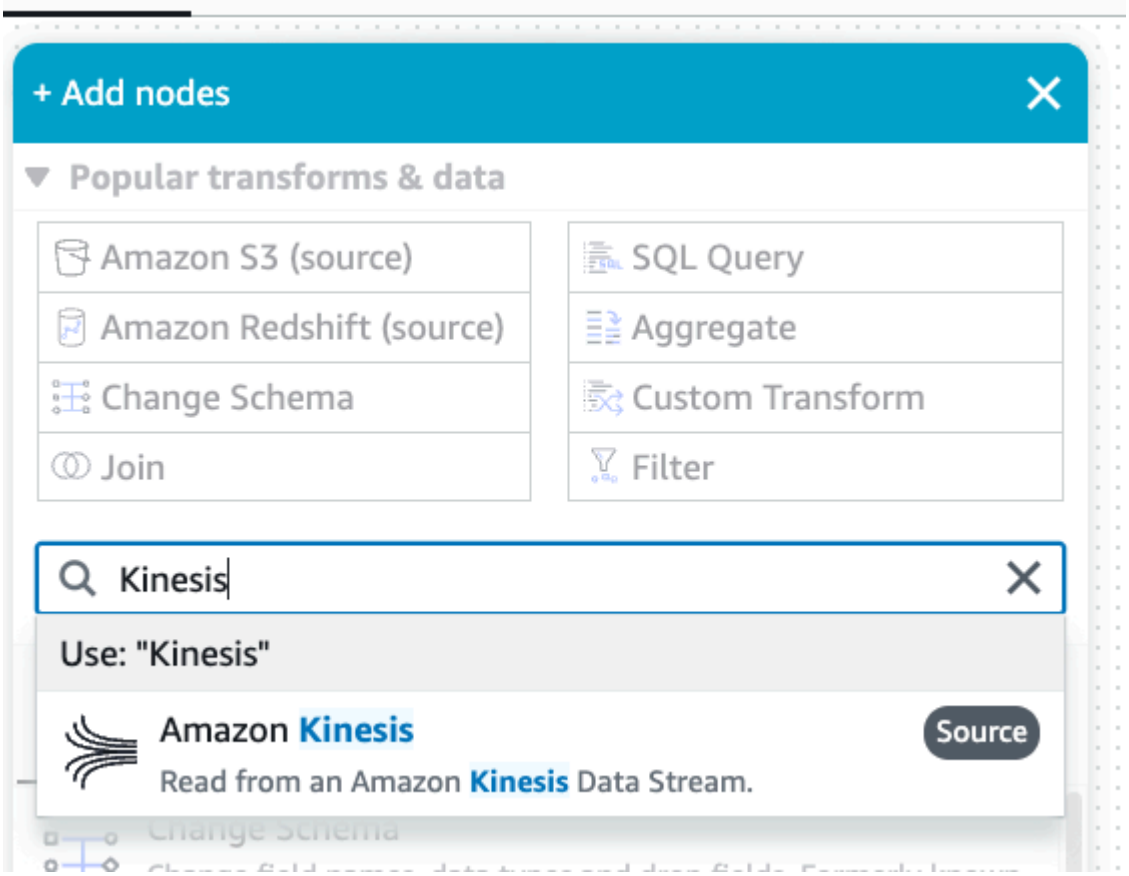


4. Navigieren Sie zur Registerkarte Auftragsdetails.
5. Geben Sie als AWS Glue Jobnamen ein`DemoStreamingJob`.
6. Wählen Sie für IAM-Rolle die von der CloudFormation Vorlage bereitgestellte Rolle aus. `glue-tutorial-role-${AWS::AccountId}`

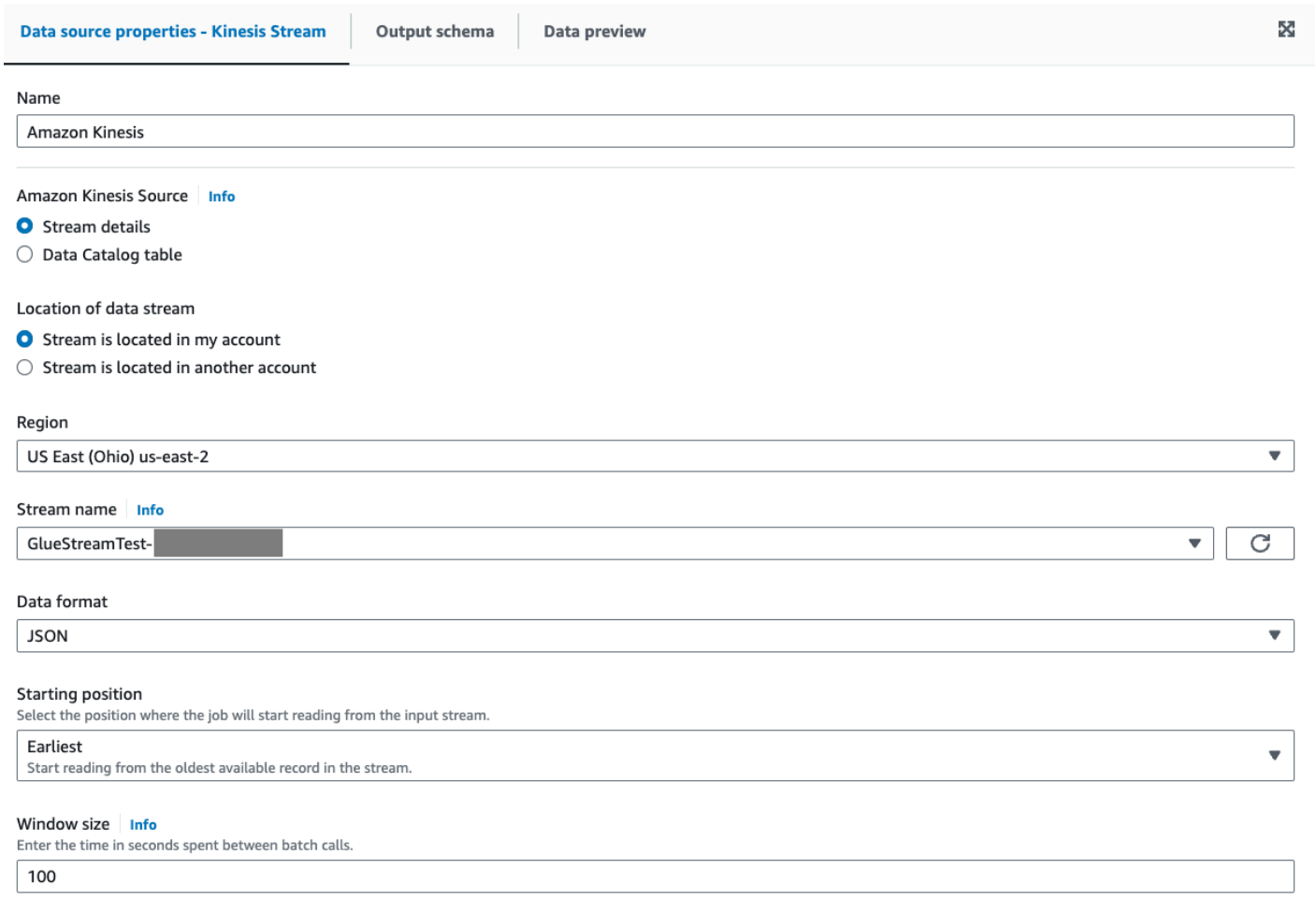
7. Wählen Sie für die Glue-Version Glue 3.0. Belassen Sie alle anderen Optionen in der Standardeinstellung.



8. Navigieren Sie zur Registerkarte Visuell.
9. Klicken Sie auf das Plus-Symbol. Geben Sie im Suchfeld Kinesis ein. Wählen Sie die Amazon-Kinesis-Datenquelle.



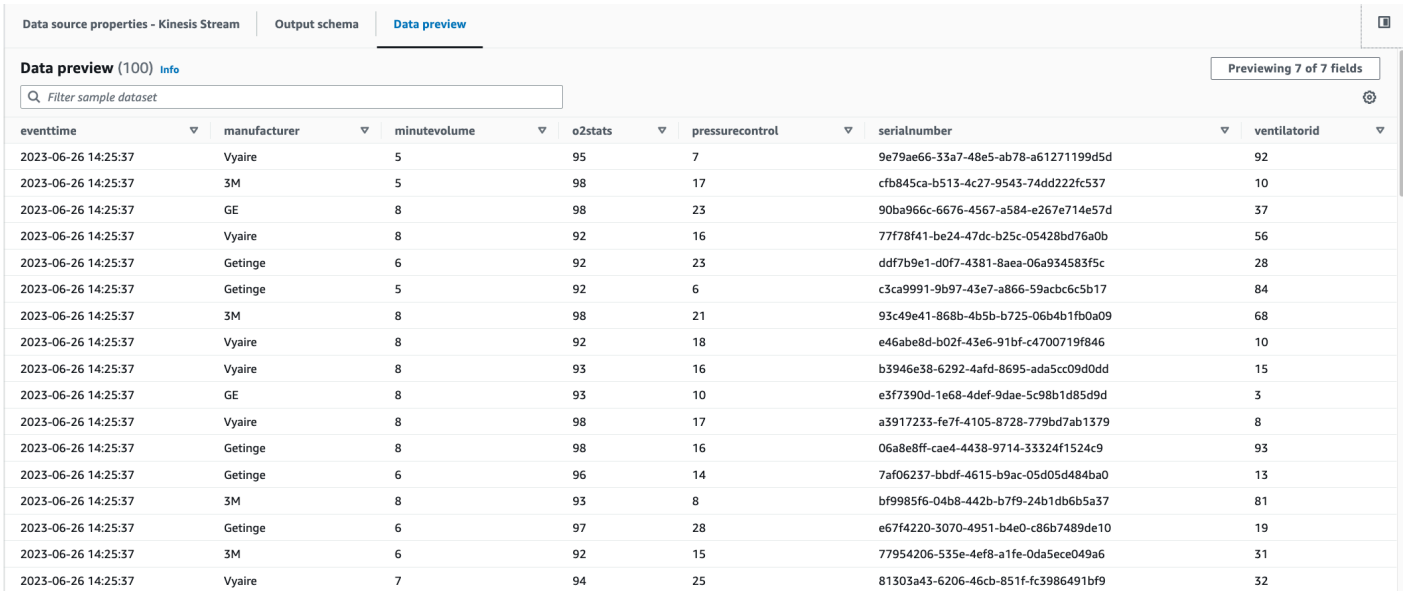
10. Wählen Sie Stream-Details für Amazon Kinesis Source unter der Registerkarte Eigenschaften der Datenquelle – Kinesis Stream.
11. Wählen Sie Stream befindet sich in meinem Konto für Speicherort des Datenstroms.
12. Wählen Sie die Region aus, die Sie verwenden.
13. Wählen Sie den `GlueStreamTest-{AWS::AccountId}`-Stream aus.
14. Behalten Sie alle anderen Einstellungen in der Standardeinstellung bei.



15. Navigieren Sie zur Registerkarte Datenvorschau.
16. Klicken Sie auf Datenvorschau-Sitzung starten, um eine Vorschau der von KDG generierten Scheindaten zu erhalten. Wählen Sie die Glue-Dienstrolle aus, die Sie zuvor für den AWS Glue Streaming-Job erstellt haben.

    Es dauert 30–60 Sekunden, bis die Vorschaudaten angezeigt werden. Wenn Keine anzuzeigenden Daten angezeigt wird, klicken Sie auf das Zahnradsymbol und ändern Sie die Anzahl der Zeilen für die Stichprobe zu `100`.

    Sie können die Beispieldaten im Folgenden sehen:



Data source properties - Kinesis Stream | Output schema | Data preview

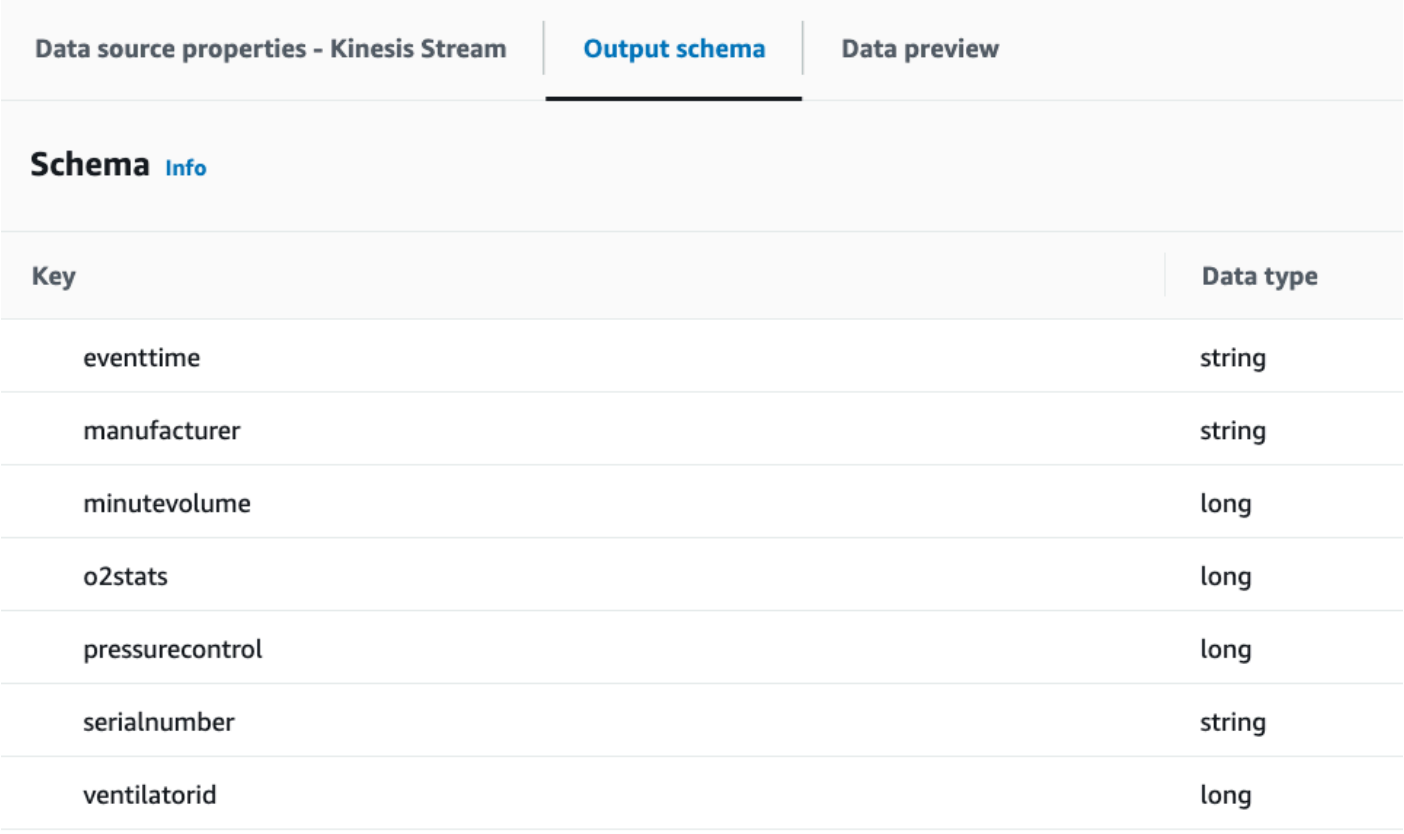
Data preview (100) Info

Previewing 7 of 7 fields

Filter sample dataset

eventtime	manufacturer	minutevolume	o2stats	pressurecontrol	serialnumber	ventilatorid
2023-06-26 14:25:37	Vyaire	5	95	7	9e79ae66-33a7-48e5-ab78-a61271199d5d	92
2023-06-26 14:25:37	3M	5	98	17	cfb845ca-b513-4c27-9543-74dd222fc537	10
2023-06-26 14:25:37	GE	8	98	23	90ba966c-6676-4567-a584-e267e714e57d	37
2023-06-26 14:25:37	Vyaire	8	92	16	77f78f41-be24-47dc-b25c-05428bd76a0b	56
2023-06-26 14:25:37	Getinge	6	92	23	ddf7b9e1-d0f7-4381-8aea-06a934583f5c	28
2023-06-26 14:25:37	Getinge	5	92	6	c3ca9991-9b97-43e7-a866-59acbc6c5b17	84
2023-06-26 14:25:37	3M	8	98	21	93c49e41-868b-4b5b-b725-06b4b1fb0a09	68
2023-06-26 14:25:37	Vyaire	8	92	18	e46abe8d-b02f-43e6-91bf-c4700719f846	10
2023-06-26 14:25:37	Vyaire	8	93	16	b3946e38-6292-4afd-8695-ada5cc09d0dd	15
2023-06-26 14:25:37	GE	8	93	10	e3f7390d-1e68-4def-9dae-5c98b1d85d9d	3
2023-06-26 14:25:37	Vyaire	8	98	17	a3917233-fe7f-4105-8728-779bd7ab1379	8
2023-06-26 14:25:37	Getinge	8	98	16	06a8e8ff-cae4-4438-9714-33324f1524c9	93
2023-06-26 14:25:37	Getinge	6	96	14	7af06237-bbdf-4615-b9ac-05d05d484ba0	13
2023-06-26 14:25:37	3M	8	93	8	bf9985f6-04b8-442b-b7f9-24b1db6b5a37	81
2023-06-26 14:25:37	Getinge	6	97	28	e67f4220-3070-4951-b4e0-c86b7489de10	19
2023-06-26 14:25:37	3M	6	92	15	77954206-535e-4ef8-a1fe-0da5ece049a6	31
2023-06-26 14:25:37	Vyaire	7	94	25	81303a43-6206-46cb-851f-fc3986491bf9	32

Sie können das abgeleitete Schema auch auf der Registerkarte Ausgabeschema sehen.



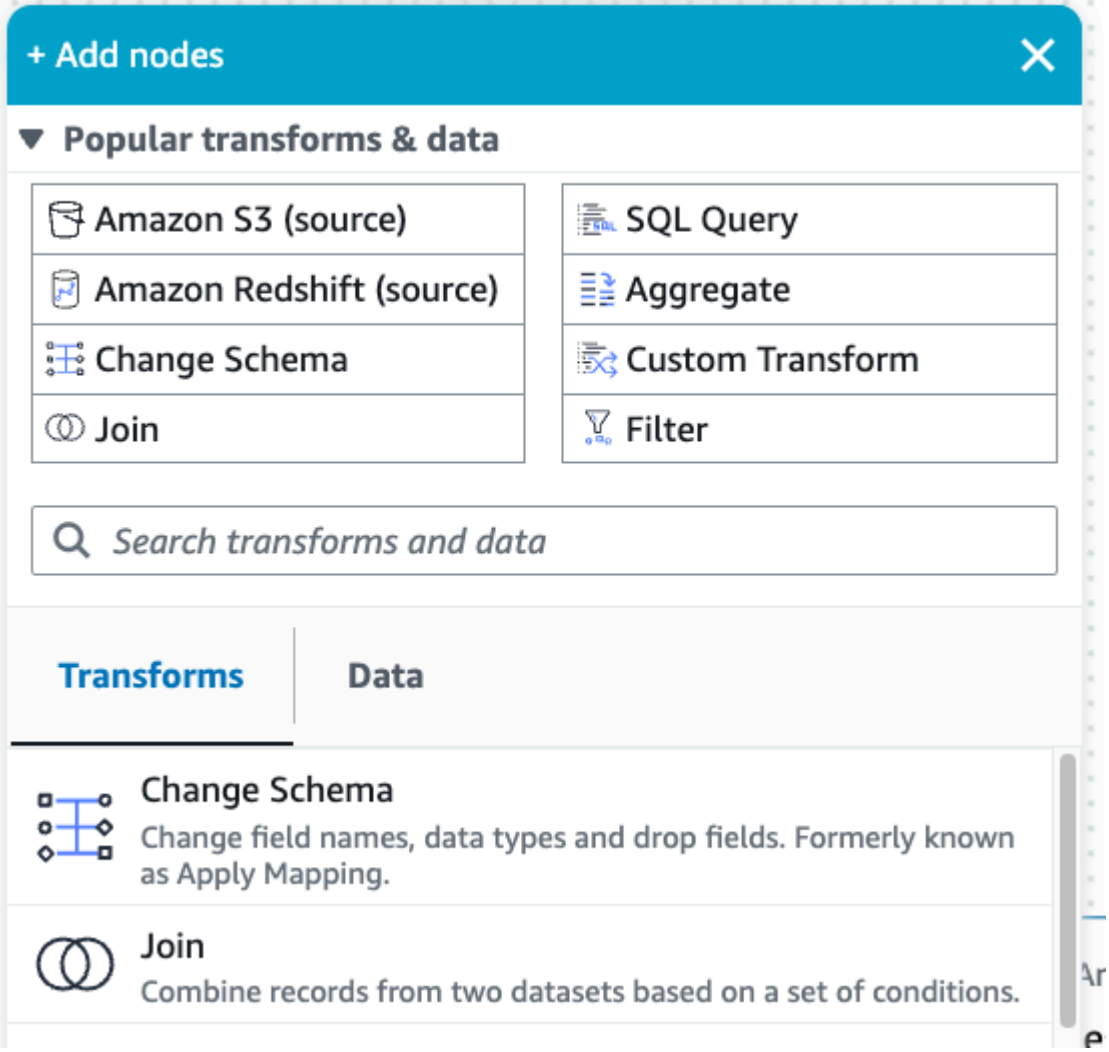
Data source properties - Kinesis Stream | Output schema | Data preview

Schema Info

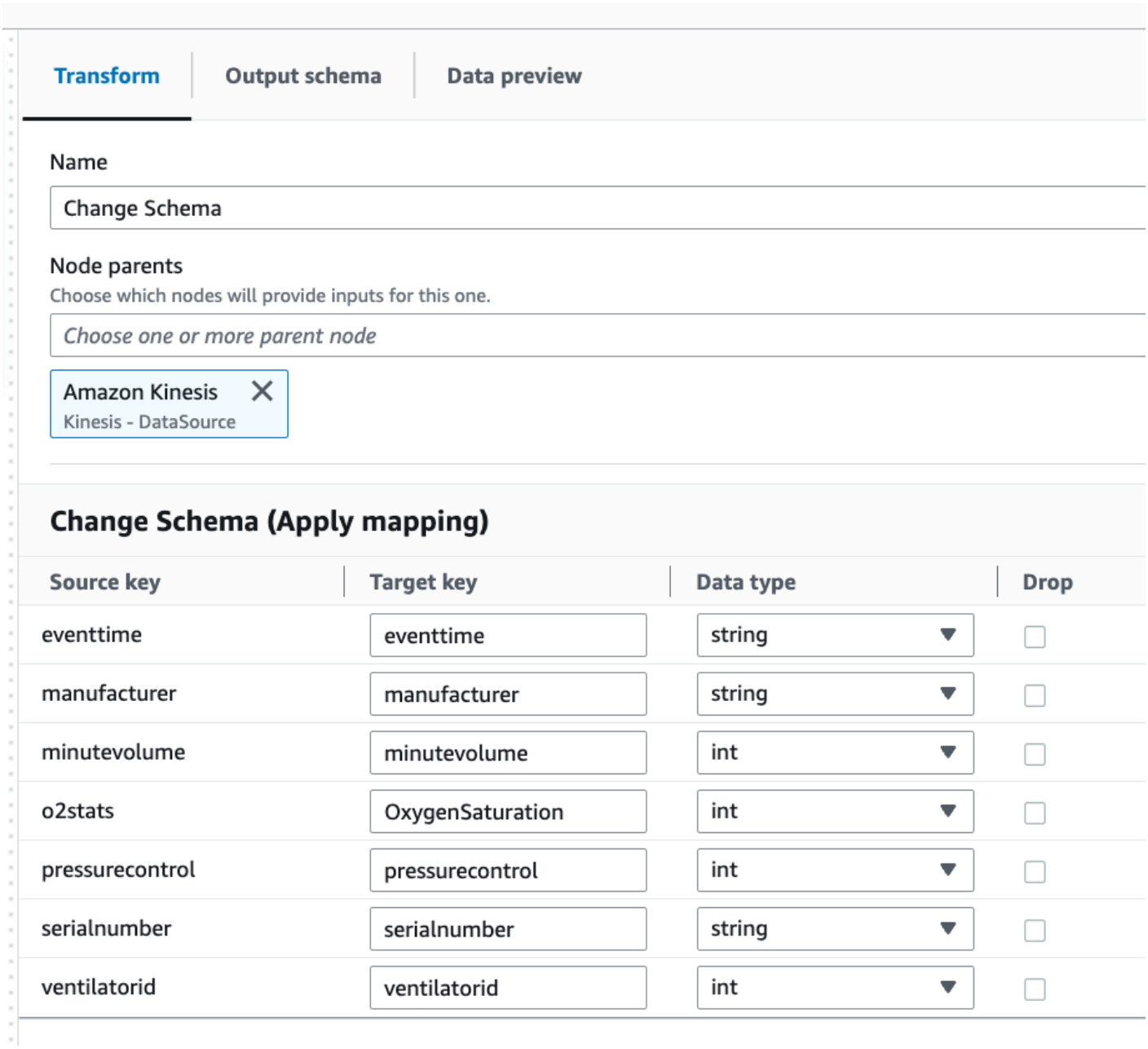
Key	Data type
eventtime	string
manufacturer	string
minutevolume	long
o2stats	long
pressurecontrol	long
serialnumber	string
ventilatorid	long

Durchführen einer Transformation und Speichern des transformierten Ergebnisses in Amazon S3

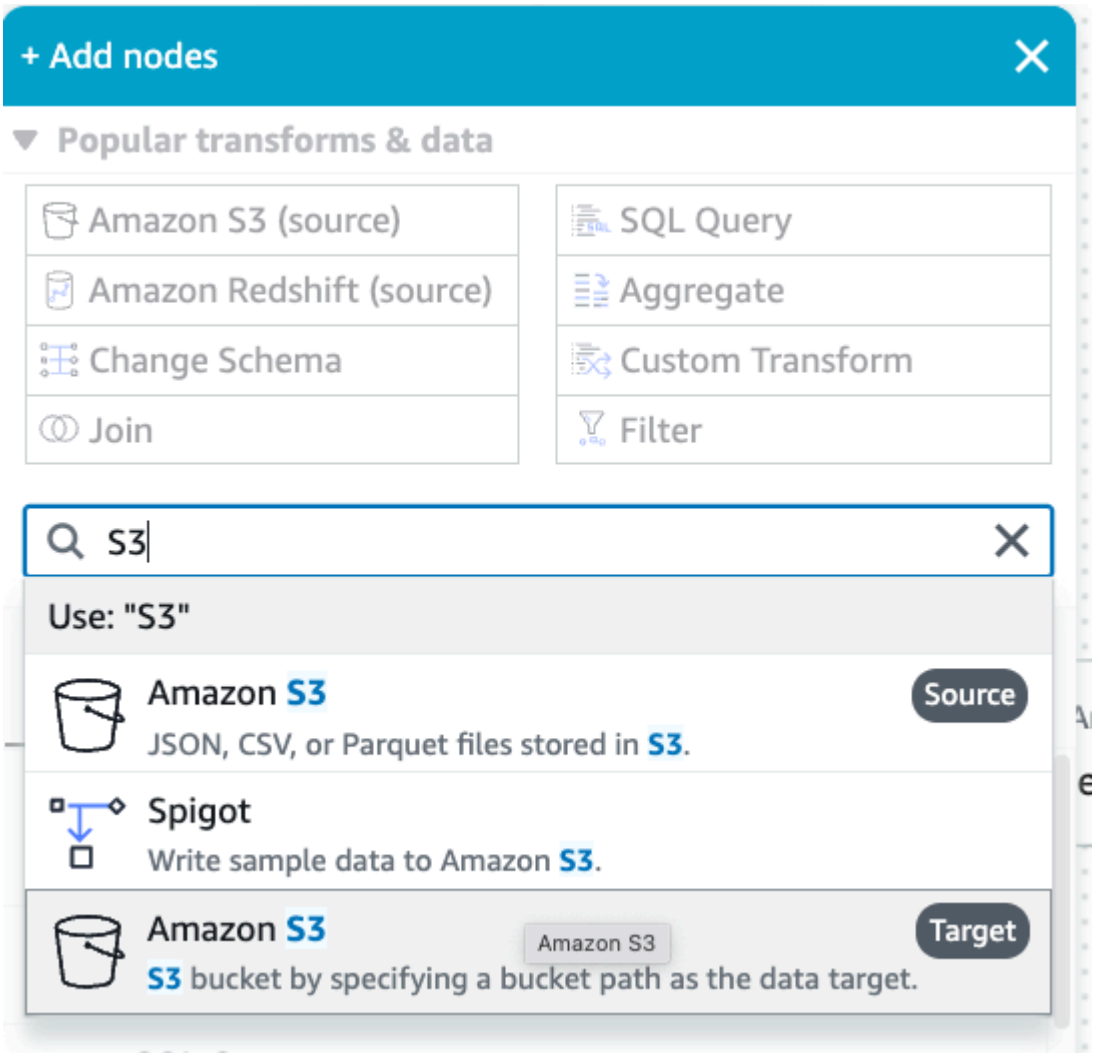
1. Klicken Sie bei ausgewähltem Quellknoten auf das Plus-Symbol oben links, um einen Transformationsschritt hinzuzufügen.
2. Wählen Sie den Schritt Schema ändern.



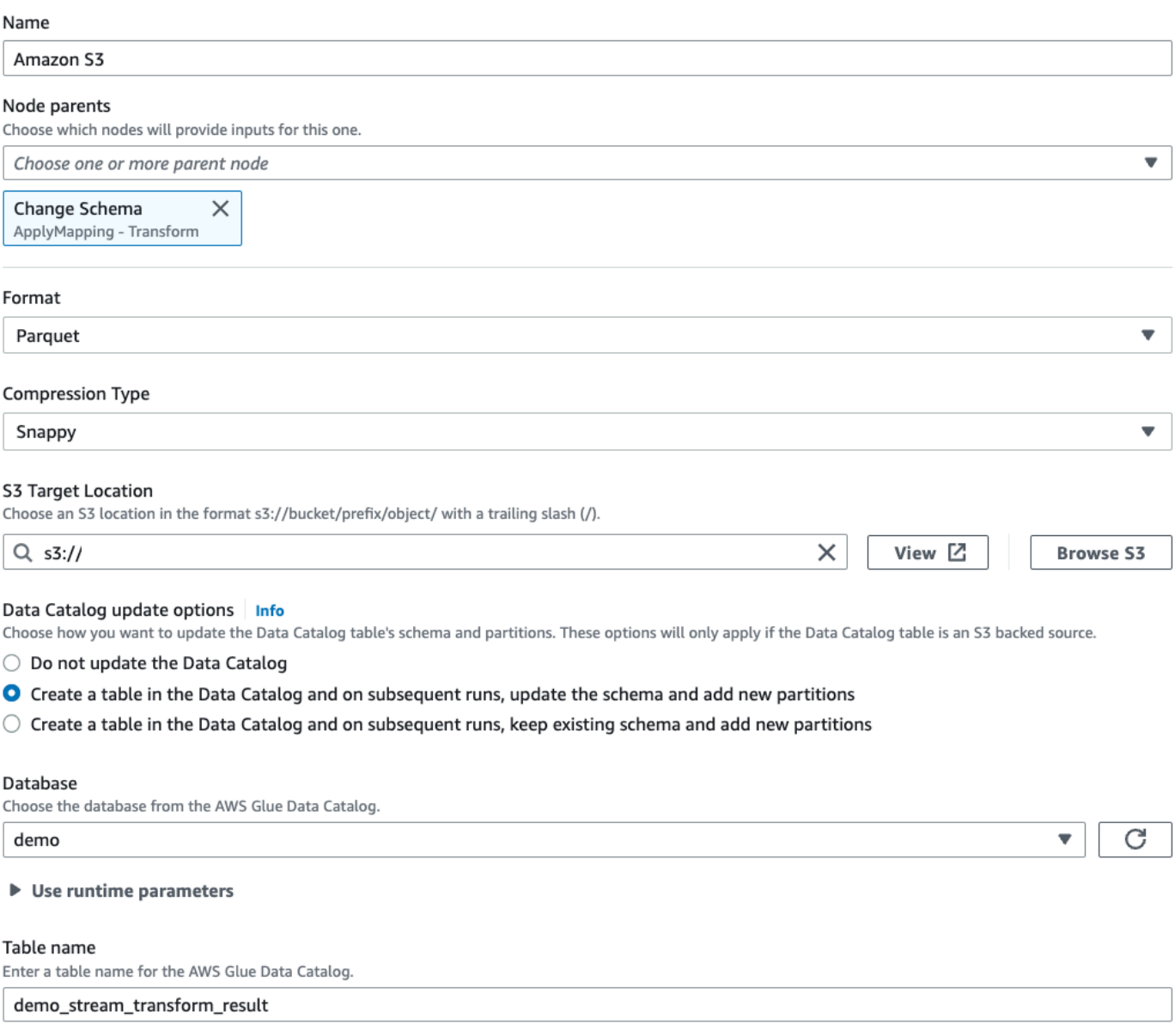
3. In diesem Schritt können Sie Felder umbenennen und den Datentyp von Feldern konvertieren. Benennen Sie die `o2stats`-Spalte zu `OxygenSaturation` um und konvertieren Sie den gesamten `long`-Datentyp zu `int`.



4. Klicken Sie auf das Plus-Symbol, um ein Amazon-S3-Ziel hinzuzufügen. Geben Sie S3 in das Suchfeld ein und wählen Sie den Schritt Amazon S3 – Zieltransformation.

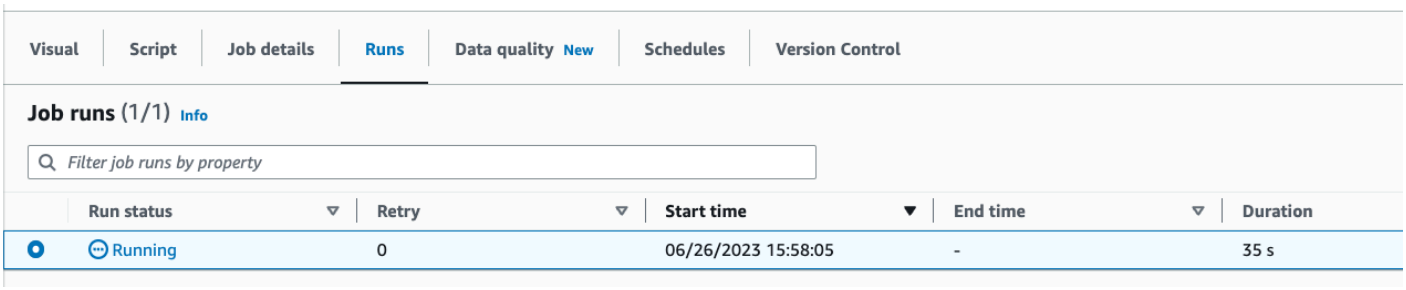


5. Wählen Sie Parquet als Zieldateiformat.
6. Wählen Sie Snappy als Komprimierungstyp.
7. Geben Sie einen mit der CloudFormation Vorlage erstellten S3-Zielort ein,`streaming-tutorial-s3-target-{AWS::AccountId}`.
8. Aktivieren Sie das Kontrollkästchen Eine Tabelle im Datenkatalog erstellen und bei späteren Ausführungen das Schema aktualisieren und neue Partitionen hinzufügen.
9. Geben Sie die Zieldatenbank und den Tabellennamen ein, um das Schema der Amazon-S3-Zieltabelle zu speichern.

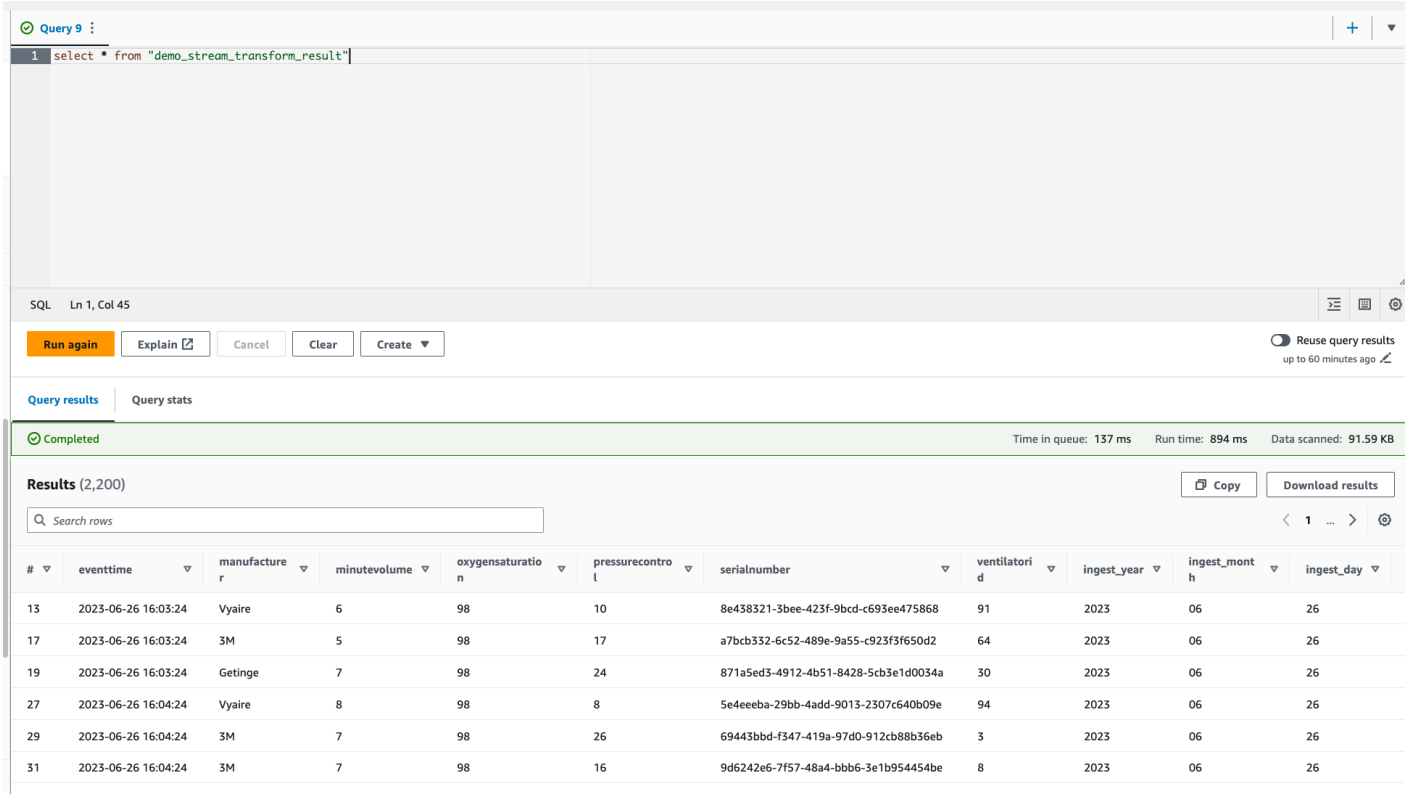


10. Klicken Sie auf die Registerkarte Skript, um den generierten Code anzuzeigen.
11. Klicken Sie oben rechts auf Speichern, um den ETL-Code zu speichern, und klicken Sie dann auf Ausführen, um den AWS Glue Streaming-Job zu starten.

    Sie finden den Ausführungsstatus auf der Registerkarte Ausführungen. Lassen Sie den Auftrag 3–5 Minuten lang laufen und stoppen Sie ihn dann.



12. Überprüfen Sie die neu erstellte Tabelle in Amazon Athena.



Tutorial: Erstellen Sie Ihren ersten Streaming-Workload mit AWS Glue Studio-Notebooks

In diesem Tutorial erfahren Sie, wie Sie AWS Glue Studio-Notebooks nutzen können, um Ihre ETL-Jobs interaktiv für die Datenverarbeitung nahezu in Echtzeit zu erstellen und zu verfeinern. Ganz gleich, ob Sie noch keine Erfahrung damit haben AWS Glue oder Ihre Fähigkeiten erweitern möchten, dieser Leitfaden führt Sie durch den Prozess und ermöglicht es Ihnen, das volle Potenzial AWS Glue interaktiver Sitzungsnotizbücher auszuschöpfen.

Mit AWS Glue Streaming können Sie Streaming-Jobs zum Extrahieren, Transformieren und Laden (ETL) erstellen, die kontinuierlich ausgeführt werden und Daten aus Streaming-Quellen wie Amazon Kinesis Data Streams, Apache Kafka und Amazon Managed Streaming for Apache Kafka (Amazon MSK) verbrauchen.

Voraussetzungen

Um diesem Tutorial zu folgen, benötigen Sie einen Benutzer mit AWS Konsolenberechtigungen zur Nutzung AWS Glue von Amazon Kinesis, Amazon S3, Amazon Athena AWS CloudFormation, AWS Lambda und Amazon Cognito.

Streaming-Daten von Amazon Kinesis verarbeiten

Themen

- Generieren von Scheindaten mit Kinesis Data Generator
- Einen AWS Glue Streaming-Job mit AWS Glue Studio erstellen
- Bereinigen
- Schlussfolgerung

Generieren von Scheindaten mit Kinesis Data Generator

> **Note**
>
> Wenn Sie unseren vorherigen Abschnitt Tutorial: Erstellen Sie Ihren ersten Streaming-Workload mit AWS Glue Studio bereits abgeschlossen haben, haben Sie den Kinesis Data Generator bereits auf Ihrem Konto installiert und können die Schritte 1–8 unten überspringen und mit Abschnitt Einen AWS Glue Streaming-Job mit AWS Glue Studio erstellen fortfahren.

Mit dem Kinesis Data Generator (KDG) können Sie Beispieldaten im JSON-Format synthetisch erzeugen. Vollständige Anweisungen und Details finden Sie in der Dokumentation des Tools.

1. Um zu beginnen, klicken Sie hier,

   Launch Stack um eine AWS CloudFormation Vorlage in Ihrer Umgebung auszuführen. AWS

> **Note**
>
> Möglicherweise tritt ein CloudFormation Vorlagenfehler auf, weil einige Ressourcen, wie z. B. der Amazon Cognito Cognito-Benutzer für Kinesis Data Generator, bereits in Ihrem AWS Konto vorhanden sind. Das könnte daran liegen, dass Sie dies bereits in einem anderen Tutorial oder Blog eingerichtet haben. Um dieses Problem zu beheben, können Sie entweder die Vorlage für einen Neuanfang in einem neuen AWS Konto ausprobieren oder eine andere AWS Region erkunden. Mit diesen Optionen können Sie das Tutorial ausführen, ohne mit bestehenden Ressourcen in Konflikt zu geraten.

Die Vorlage stellt einen Kinesis-Datenstrom und ein Kinesis-Data-Generator-Konto für Sie bereit.

2. Geben Sie einen Benutzernamen und ein Passwort ein, mit denen sich der KDG authentifizieren soll. Notieren Sie sich den Benutzernamen und das Passwort für die weitere Verwendung.
3. Wählen Sie Weiter bis zum letzten Schritt. Bestätigen Sie die Erstellung von IAM-Ressourcen. Suchen Sie oben auf dem Bildschirm nach Fehlern, z. B. wenn das Passwort nicht den Mindestanforderungen entspricht, und stellen Sie die Vorlage bereit.
4. Navigieren Sie zur Registerkarte Ausgaben des Stacks. Sobald die Vorlage bereitgestellt ist, wird die generierte Eigenschaft angezeigt KinesisDataGeneratorUrl. Klicken Sie auf diese URL.
5. Geben Sie den Benutzernamen und das Passwort ein, die Sie sich notiert haben.
6. Wählen Sie die Region aus, die Sie verwenden, und wählen Sie den Kinesis-Stream `GlueStreamTest-{AWS::AccountId}`
7. Geben Sie die folgende Vorlage ein:

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
            "min":5,
            "max":30
        }
    )}},
    "o2stats": {{random.number(
        {
            "min":92,
```

```
            "max":98
        }
    )}},
    "minutevolume": {{random.number(
        {
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Sie können jetzt Scheindaten mit der Testvorlage anzeigen und die Scheindaten mit Daten senden in Kinesis aufnehmen.

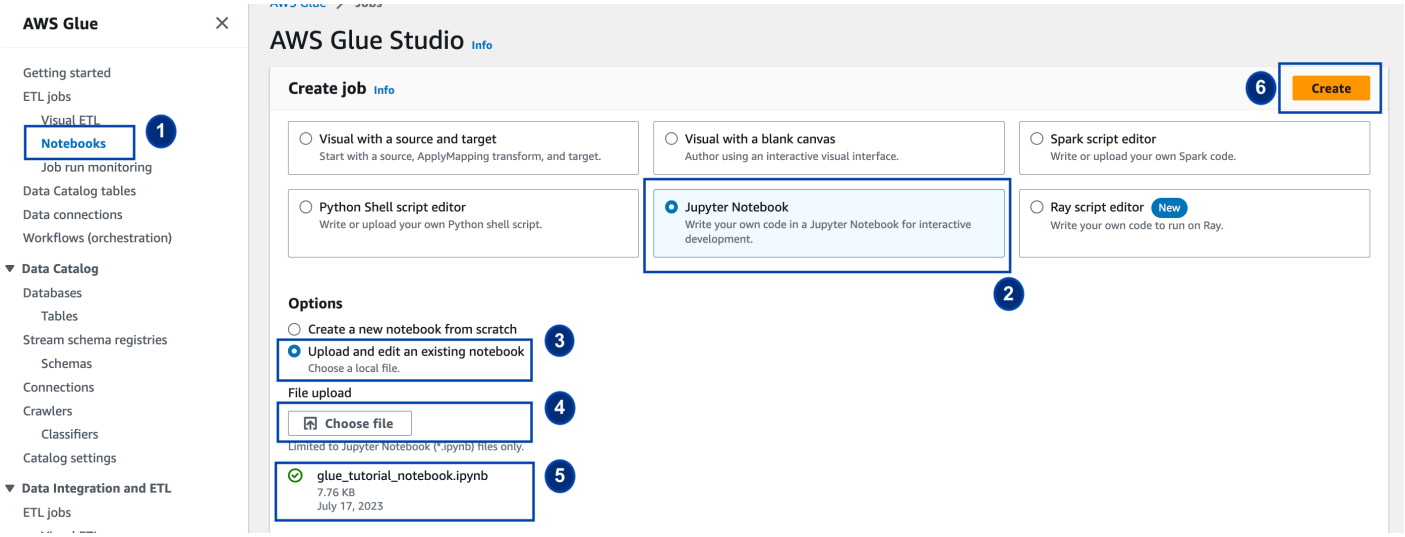
8. Klicken Sie auf Daten senden und generieren Sie 5–10 000 Datensätze für Kinesis.

Einen AWS Glue Streaming-Job mit AWS Glue Studio erstellen

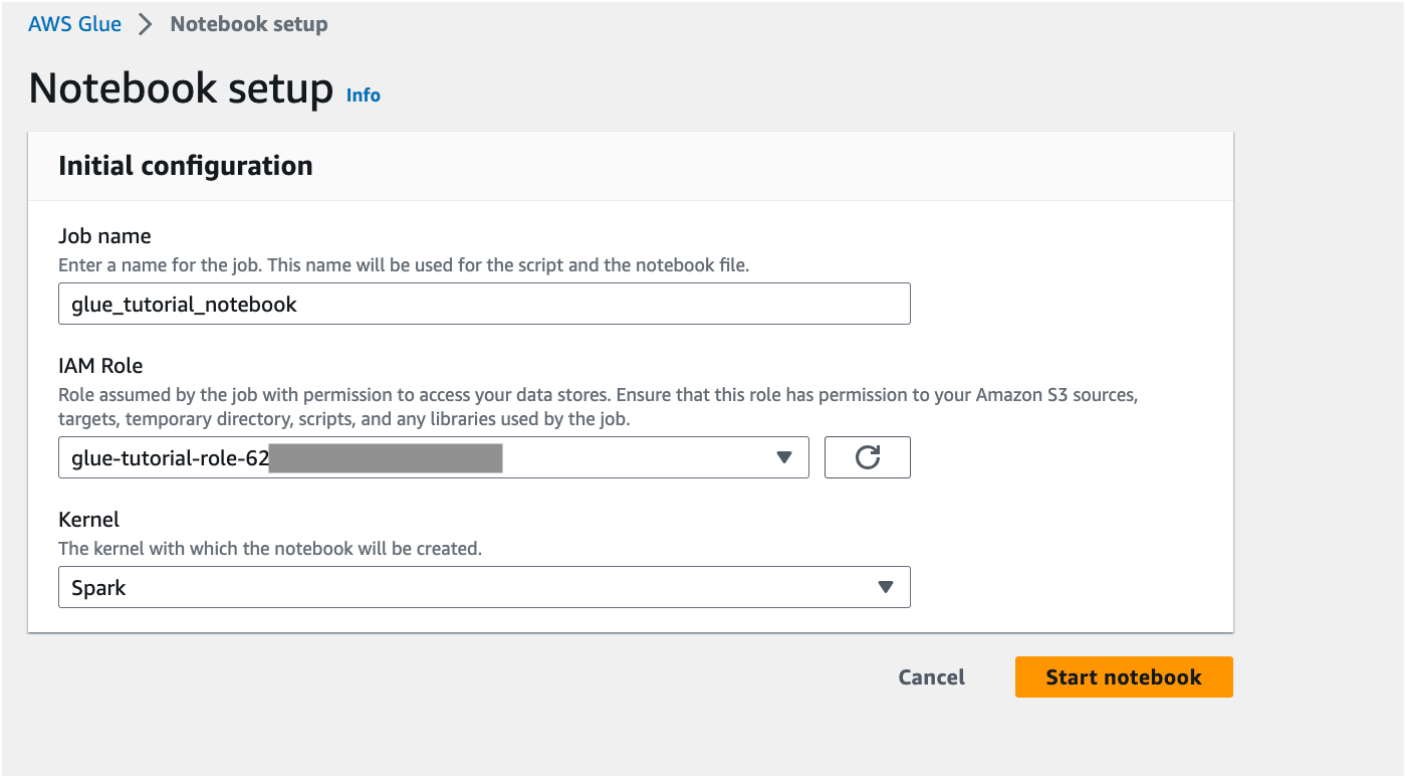
AWS Glue Studio ist eine visuelle Oberfläche, die das Entwerfen, Orchestrieren und Überwachen von Datenintegrationspipelines vereinfacht. Es ermöglicht Benutzern die Erstellung von Datenumwandlungs-Pipelines, ohne umfangreichen Code schreiben zu müssen. Neben dem visuellen Authoring-Erlebnis bietet AWS Glue Studio auch ein Jupyter-Notizbuch mit AWS Glue interaktiven Sessions, das Sie im weiteren Verlauf dieses Tutorials verwenden werden.

Richten Sie den Job „Interaktive Streaming-Sitzungen AWS Glue “ ein

1. Die bereitgestellte Notebook-Datei herunterladen und sie in einem lokalen Verzeichnis speichern
2. Öffnen Sie die AWS Glue Konsole und klicken Sie im linken Bereich auf Notizbücher > Jupyter Notebook > Ein vorhandenes Notizbuch hochladen und bearbeiten. Laden Sie das Notebook aus dem vorherigen Schritt hoch und klicken Sie auf Erstellen.



3. Geben Sie dem Auftrag einen Namen und eine Rolle und wählen Sie den Standard-Spark-Kernel. Klicken Sie anschließend auf Notebook starten. Wählen Sie für die IAM-Rolle die Rolle aus, die durch die Vorlage bereitgestellt wird. CloudFormation Sie können dies auf der Registerkarte Ausgaben von sehen. CloudFormation

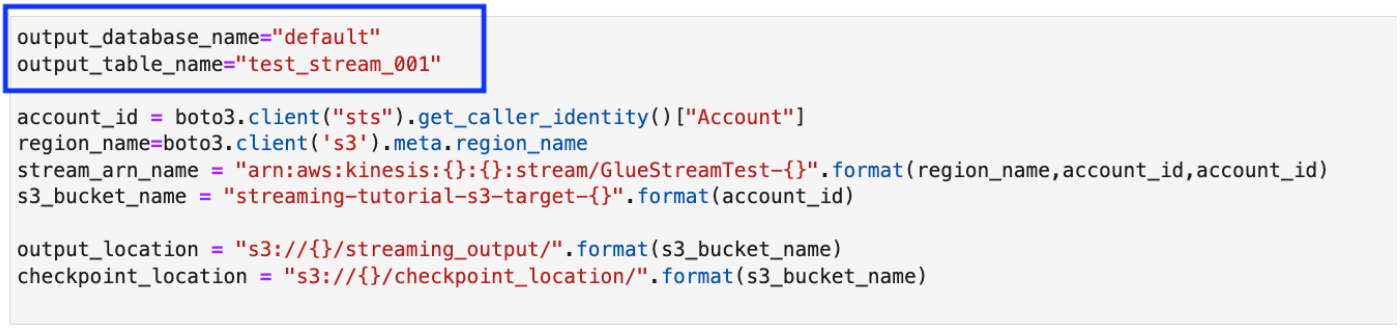


Das Notebook enthält alle notwendigen Anweisungen, um das Tutorial fortzusetzen. Sie können entweder die Anweisungen auf dem Notebook ausführen oder diesem Tutorial folgen, um mit der Auftragsentwicklung fortzufahren.

Die Notebook-Zellen ausführen

1. (Optional) Die erste Codezelle, `%help` listet alle verfügbaren Notebook-Magics auf. Sie können diese Zelle vorerst überspringen, aber Sie können sie auch gerne erkunden.
2. Beginnen Sie mit dem nächsten Codeblock, `%streaming`. Diese Magie legt den Jobtyp auf Streaming fest, sodass Sie einen AWS Glue Streaming-ETL-Job entwickeln, debuggen und bereitstellen können.
3. Führen Sie die nächste Zelle aus, um eine AWS Glue interaktive Sitzung zu erstellen. Die Ausgabezelle enthält eine Nachricht, die die Erstellung der Sitzung bestätigt.

4. In der nächsten Zelle werden die Variablen definiert. Ersetzen Sie die Werte durch solche, die für Ihren Auftrag geeignet sind, und führen Sie die Zelle aus. Zum Beispiel:



5. Da die Daten bereits zu Kinesis Data Streams gestreamt werden, wird Ihre nächste Zelle die Ergebnisse aus dem Stream verarbeiten. Führen Sie die nächste Zelle aus. Da es keine Druckanweisungen gibt, wird von dieser Zelle keine Ausgabe erwartet.
6. In der folgenden Zelle erkunden Sie den eingehenden Stream, indem Sie einen Mustersatz nehmen und dessen Schema und die tatsächlichen Daten ausgeben. Zum Beispiel:

Sample and print the incoming records

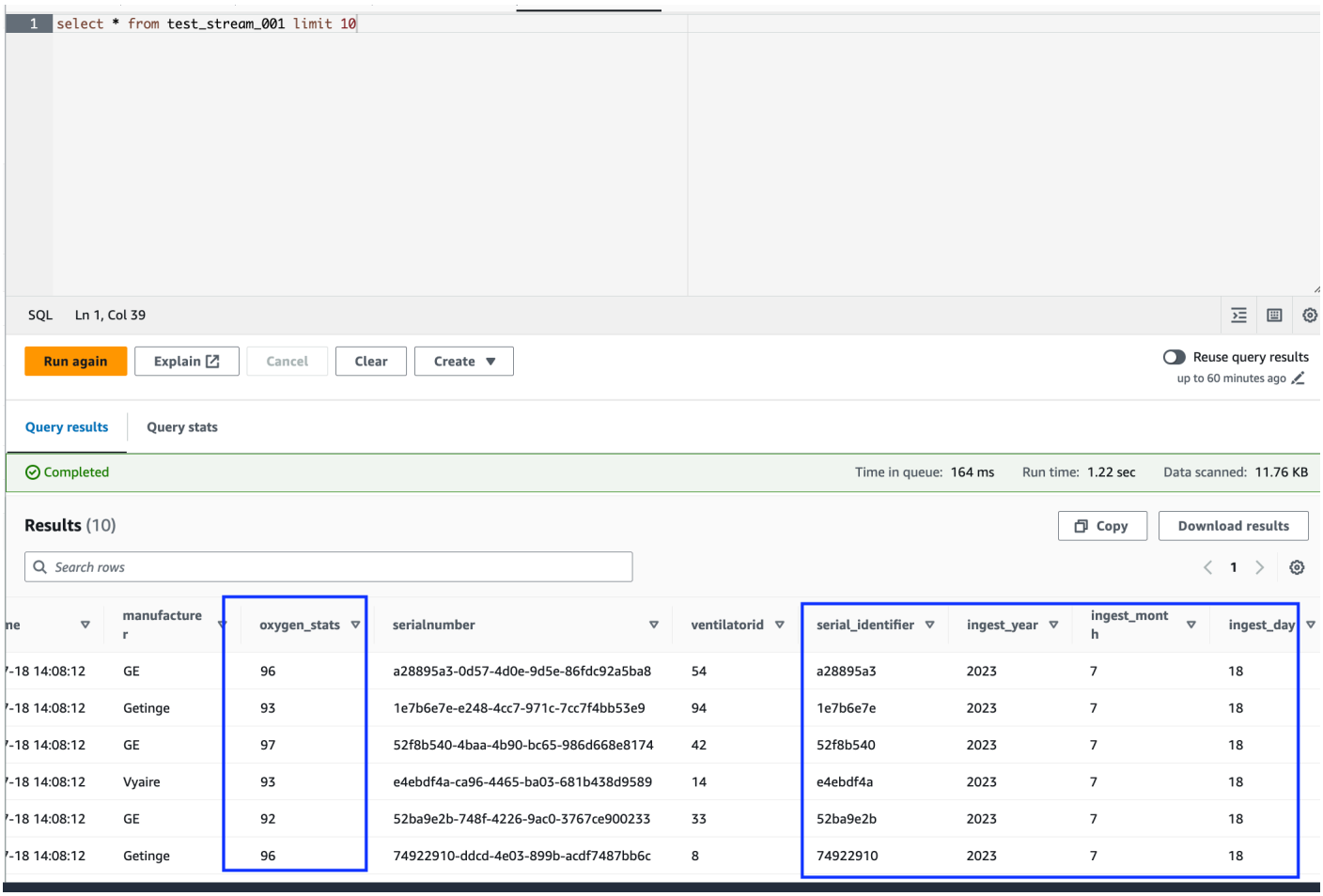
the sampling is for debugging purpose. You may comment off the entire code cell below, before deploying the actual code

```
[4]: options = {
       "pollingTimeInMs": "20000",
       "windowSize": "5 seconds"
     }
     sampled_dynamic_frame = glueContext.getSampleStreamingDynamicFrame(data_frame, options, None)

     count_of_sampled_records = sampled_dynamic_frame.count()

     print(count_of_sampled_records)

     sampled_dynamic_frame.printSchema()

     sampled_dynamic_frame.toDF().show(10,False)
```

```
100
root
|-- eventtime: string
|-- manufacturer: string
|-- minutevolume: long
|-- o2stats: long
|-- pressurecontrol: long
|-- serialnumber: string
|-- ventilatorid: long

+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|eventtime          |manufacturer|minutevolume|o2stats|pressurecontrol|serialnumber                        |ventilatorid|
+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|2023-07-18 10:20:11|3M          |6           |92     |24             |a3e860ba-24b9-41c4-bc10-91c6b35e1406|6           |
|2023-07-18 10:20:11|Vyaire      |6           |95     |6              |96101dca-3e88-457f-b390-e3291df48a81|26          |
|2023-07-18 10:20:12|Getinge     |8           |96     |24             |18f3d448-1dee-4c80-835b-1a0daa818915|22          |
|2023-07-18 10:20:12|Getinge     |7           |98     |30             |25f425cd-b978-4953-9a03-4d607a639364|91          |
|2023-07-18 10:20:12|GE          |5           |93     |25             |2cd7cdc2-f5f5-4ff2-ae32-45e5a8922d53|93          |
```

7. Als nächstes definieren Sie die eigentliche Datentransformationslogik. Die Zelle besteht aus der `processBatch`-Methode, die bei jedem Micro-Batch ausgelöst wird. Führen Sie die Zelle aus. Im Großen und Ganzen machen wir Folgendes mit dem eingehenden Stream:
   a. Wählen Sie eine Teilmenge der Eingabespalten aus.
   b. Benennen Sie eine Spalte um (o2stats in oxygen_stats).
   c. Leiten Sie neue Spalten ab (serial_identifier, ingest_year, ingest_month und ingest_day).
   d. Speichern Sie die Ergebnisse in einem Amazon S3 S3-Bucket und erstellen Sie auch eine partitionierte AWS Glue Katalogtabelle
8. In der letzten Zelle lösen Sie den Prozess-Batch alle 10 Sekunden aus. Führen Sie die Zelle aus und warten Sie etwa 30 Sekunden, bis sie den Amazon S3 S3-Bucket und die AWS Glue Katalogtabelle gefüllt hat.

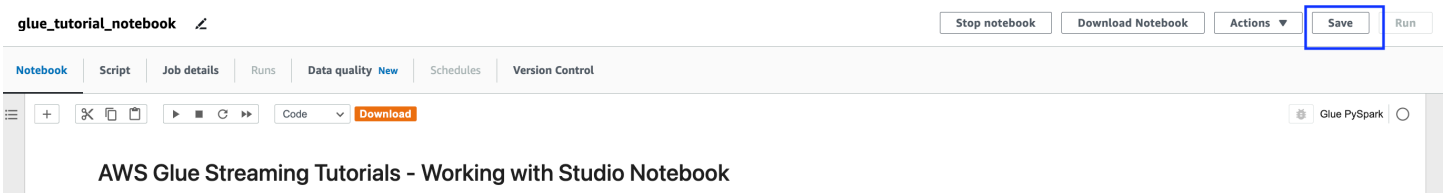
9. Schließlich durchsuchen Sie die gespeicherten Daten mit dem Abfrage-Editor von Amazon Athena. Sie können die umbenannte Spalte und auch die neuen Partitionen sehen.



Das Notebook enthält alle notwendigen Anweisungen, um das Tutorial fortzusetzen. Sie können entweder die Anweisungen auf dem Notebook ausführen oder diesem Tutorial folgen, um mit der Auftragsentwicklung fortzufahren.

Speichern Sie den Job und führen Sie ihn aus AWS Glue

Wenn die Entwicklung und das Testen Ihrer Anwendung mit dem Notebook für interaktive Sitzungen abgeschlossen sind, klicken Sie oben auf der Oberfläche des Notebooks auf Speichern. Nach dem Speichern können Sie die Anwendung auch als Auftrag ausführen.



Bereinigen

Um zusätzliche Kosten für Ihr Konto zu vermeiden, stoppen Sie den Streaming-Auftrag, den Sie im Rahmen des Tutorials gestartet haben. Sie können dies tun, indem Sie das Notebook anhalten, wodurch die Sitzung beendet wird. Leeren Sie den Amazon S3 S3-Bucket und löschen Sie den AWS CloudFormation Stack, den Sie zuvor bereitgestellt haben.

Schlussfolgerung

In diesem Tutorial haben wir gezeigt, wie Sie mit dem AWS Glue Studio-Notizbuch Folgendes tun können

- Einen Streaming-ETL-Auftrag mithilfe von Notebooks erstellen
- Eine Vorschau eingehender Datenströme anzeigen
- Codieren und beheben Sie Probleme, ohne AWS Glue Jobs veröffentlichen zu müssen
- Überprüfen Sie den end-to-end funktionierenden Code, entfernen Sie jegliches Debugging und drucken Sie Anweisungen oder Zellen aus dem Notizbuch
- Veröffentlichen Sie den Code als Job AWS Glue

Das Ziel dieses Tutorials ist es, Ihnen praktische Erfahrungen mit AWS Glue Streaming und interaktiven Sessions zu vermitteln. Wir empfehlen Ihnen, dies als Referenz für Ihre individuellen AWS Glue Streaming-Anwendungsfälle zu verwenden. Weitere Informationen finden Sie unter Erste Schritte mit AWS Glue interaktive Sessions.

AWS Glue Streaming-Konzepte

Die folgenden Abschnitte enthalten Informationen zu AWS Glue Streaming-Konzepten.

Themen

- Aufbau eines AWS Glue Streaming-Jobs

Aufbau eines AWS Glue Streaming-Jobs

AWS Glue Streaming-Jobs basieren auf dem Spark-Streaming-Paradigma und nutzen strukturiertes Streaming aus dem Spark-Framework. Streaming-Aufträge fragen in einem bestimmten Zeitintervall

ständig die Streaming-Datenquelle ab, um Datensätze als Micro-Batches abzurufen. In den folgenden Abschnitten werden die verschiedenen Teile eines AWS Glue Streaming-Jobs untersucht.

forEachBatch

Die `forEachBatch` Methode ist der Einstiegspunkt eines ausgeführten Streaming-Jobs. AWS Glue AWS Glue Streaming-Jobs verwenden `forEachBatch` diese Methode, um Daten abzufragen. Sie funktioniert wie ein Iterator, der während des gesamten Lebenszyklus des Streaming-Jobs aktiv bleibt, die Streaming-Quelle regelmäßig nach neuen Daten abfragt und die neuesten Daten in Mikrobatches verarbeitet.

```
glueContext.forEachBatch(
    frame=dataFrame_AmazonKinesis_node1696872487972,
    batch_function=processBatch,
    options={
        "windowSize": "100 seconds",
        "checkpointLocation": args["TempDir"] + "/" + args["JOB_NAME"] + "/
checkpoint/",
    },
)
```

Konfigurieren Sie die Eigenschaft `frame` von `forEachBatch`, um eine Streaming-Quelle anzugeben. In diesem Beispiel wird der Quellknoten, den Sie bei der Auftragserstellung auf der leeren Arbeitsfläche erstellt haben, mit der Standardeinstellung DataFrame des Jobs gefüllt. Legen Sie die Eigenschaft `batch_function` als die `function` fest, das Sie für jede Micro-Batch-Operation aufrufen möchten. Sie müssen eine Funktion definieren, die die Batch-Transformation der eingehenden Daten übernimmt.

Quelle

Im ersten Schritt der `processBatch` Funktion überprüft das Programm die Anzahl der Datensätze der Eigenschaft DataFrame , die Sie als Frame definiert haben. `forEachBatch` Das Programm fügt einem Wert, der nicht leer ist, einen Zeitstempel für die Aufnahme an. DataFrame Die `data_frame.count()>0`-Klausel bestimmt, ob der letzte Micro-Batch nicht leer ist und für die weitere Verarbeitung bereit ist.

```
def processBatch(data_frame, batchId):
    if data_frame.count() >0:
        AmazonKinesis_node1696872487972 = DynamicFrame.fromDF(
            glueContext.add_ingestion_time_columns(data_frame, "hour"),
            glueContext,
            "from_data_frame",
        )
```

Mapping (Zuordnung)

Im nächsten Abschnitt des Programms wird Mapping angewendet. Die `Mapping.apply` Methode auf einem Spark DataFrame ermöglicht es Ihnen, Transformationsregeln für Datenelemente zu definieren. Normalerweise können Sie die Quelldatenspalte umbenennen, den Datentyp ändern oder eine benutzerdefinierte Funktion darauf anwenden und diese den Zielspalten zuordnen.

```
    #Script generated for node ChangeSchema
    ChangeSchema_node16986872679326 = ApplyMapping.apply(
        frame =  AmazonKinesis_node1696872487972,
        mappings = [
            ("eventtime", "string", "eventtime", "string"),
            ("manufacturer", "string", "manufacturer", "string"),
```

```
            ("minutevolume", "long", "minutevolume", "int"),
            ("o2stats", "long", "OxygenSaturation", "int"),
            ("pressurecontrol", "long", "pressurecontrol", "int"),
            ("serialnumber", "string", "serialnumber", "string"),
            ("ventilatorid", "long", "ventilatorid", "long"),
            ("ingest_year", "string", "ingest_year", "string"),
            ("ingest_month", "string", "ingest_month", "string"),
            ("ingest_day", "string", "ingest_day", "string"),
            ("ingest_hour", "string", "ingest_hour", "string"),
        ],
        transformation_ctx="ChangeSchema_node16986872679326",
    )
        )
```

Sink

In diesem Abschnitt werden die von der Streaming-Quelle eingehenden Datensätze an einem Zielort gespeichert. In diesem Beispiel werden wir die Daten in einen Amazon-S3-Speicherort schreiben. Die `AmazonS3_node_path`-Eigenschaftsdetails werden entsprechend den Einstellungen, die Sie bei der Auftragserstellung in der Vorlage verwendet haben, vorausgefüllt. Sie können `updateBehavior` auf der Grundlage Ihres Anwendungsfalls einstellen und entscheiden, ob Sie die Datenkatalogtabelle nicht aktualisieren, einen Datenkatalog erstellen und das Datenkatalogschema bei nachfolgenden Ausführungen aktualisieren oder eine Katalogtabelle erstellen und die Schemadefinition bei nachfolgenden Ausführungen nicht aktualisieren.

Die Eigenschaft `partitionKeys` definiert die Speicherpartitionsoption. Standardmäßig werden die Daten gemäß dem im Quellabschnitt zur Verfügung gestellten `ingestion_time_columns` aufgeteilt. Mit dieser `compression`-Eigenschaft können Sie den Komprimierungsalgorithmus festlegen, der beim Schreiben des Ziels angewendet werden soll. Sie haben die Möglichkeit, Snappy, LZO oder GZIP als Komprimierungstechnik einzustellen. Die `enableUpdateCatalog`-Eigenschaft bestimmt, ob die AWS Glue -Katalogtabelle aktualisiert werden muss. Verfügbare Optionen für diese Eigenschaft sind `True` oder `False`.

```
    #Script generated for node Amazon S3
    AmazonS3_node1696872743449 = glueContext.getSink(
        path =  AmazonS3_node1696872743449_path,
        connection_type = "s3",
        updateBehavior = "UPDATE_IN_DATABASE",
        partitionKeys = ["ingest_year", "ingest_month", "ingest_day", "ingest_hour"],
```

```
        compression = "snappy",
        enableUpdateCatalog = True,
        transformation_ctx = "AmazonS3_node1696872743449",
    )
```

AWS Glue Katalogsenke

Dieser Abschnitt des Jobs steuert das Verhalten bei der Aktualisierung der AWS Glue Katalogtabelle. Satz `catalogDatabase` und `catalogTableName` Eigenschaft entsprechend Ihrem AWS Glue Katalogdatenbanknamen und dem Tabellennamen, der mit dem AWS Glue Job verknüpft ist, den Sie entwerfen. Sie können das Dateiformat der Zieldaten über die `setFormat`-Eigenschaft festlegen. In diesem Beispiel werden wir die Daten im Parquet-Format speichern.

Sobald Sie den AWS Glue Streaming-Job eingerichtet und ausgeführt haben, der auf dieses Tutorial verweist, Amazon Kinesis Data Streams werden die Streaming-Daten, die unter erzeugt wurden, am Amazon S3 S3-Standort in einem Parquet-Format mit schneller Komprimierung gespeichert. Bei erfolgreichen Ausführungen des Streaming-Auftrags können Sie die Daten über Amazon Athena abfragen.

```
    AmazonS3_node1696872743449 = setCatalogInfo(
        catalogDatabase =  "demo", catalogTableName = "demo_stream_transform_result"
    )
    AmazonS3_node1696872743449.setFormat("glueparquet")
    AmazonS3_node1696872743449.writeFormat("ChangeSchema_node16986872679326")
    )
```

AWS Glue Streaming-Verbindungen

Die folgenden Abschnitte enthalten Informationen zur Verwendung von Verbindungen beim AWS Glue Streaming.

Themen

- Arbeiten mit Kafka-Verbindungen
- Mit Kinesis-Verbindungen arbeiten

Arbeiten mit Kafka-Verbindungen

Sie können eine Kafka-Verbindung verwenden, um Kafka-Datenströme zu lesen und in sie zu schreiben, indem Sie Informationen verwenden, die in einer Datenkatalogtabelle gespeichert sind, oder indem Sie Informationen für den direkten Zugriff auf den Datenstrom bereitstellen. Die Verbindung unterstützt einen Kafka-Cluster oder einen Amazon Managed Streaming for Apache Kafka Kafka-Cluster. Sie können Informationen aus Kafka in einen Spark einlesen DataFrame und sie dann in einen AWS Glue DynamicFrame umwandeln. Sie können in einem JSON-Format in Kafka schreiben DynamicFrames . Wenn Sie direkt auf den Datenstrom zugreifen, verwenden Sie diese Optionen, um Informationen zum Zugriff auf den Datenstrom bereitzustellen.

Wenn Sie Datensätze aus einer Kafka-Streaming-Quelle verwenden `getCatalogSource` oder `create_data_frame_from_catalog` `getCatalogSink` oder `write_dynamic_frame_from_catalog` um Datensätze in Kafka zu schreiben, und der Job über die Datenkatalogdatenbank und die Tabellennameninformationen verfügt und diese verwenden kann, um einige grundlegende Parameter für das Lesen aus der Kafka-Streaming-Quelle abzurufen. Wenn Sie`getSource`,,`getCatalogSink`, `createDataFrameFromOptions` oder `getSourceWithFormatgetSinkWithFormat`, oder verwenden`create_data_frame_from_options`, müssen Sie diese grundlegenden Parameter mithilfe der hier beschriebenen Verbindungsoptionen angeben. `write_dynamic_frame_from_catalog`

Sie können die Verbindungsoptionen für Kafka mithilfe der folgenden Argumente für die angegebenen Methoden in der `GlueContext` Klasse angeben.

- Scala
  - `connectionOptions`: mit `getSource`, `createDataFrameFromOptions`, `getSink` verwenden
  - `additionalOptions`: mit `getCatalogSource`, `getCatalogSink` verwenden
  - `options`: mit `getSourceWithFormat`, `getSinkWithFormat` verwenden
- Python
  - `connection_options`: mit `create_data_frame_from_options`, `write_dynamic_frame_from_options` verwenden
  - `additional_options`: mit `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog` verwenden
  - `options`: mit `getSource`, `getSink` verwenden

Hinweise und Einschränkungen zum Streaming von ETL-Aufträgen finden Sie unter the section called “Hinweise zu und Einschränkungen für Streaming-ETL”.

Kafka konfigurieren

Es gibt keine AWS Voraussetzungen, um eine Verbindung zu Kafka-Streams herzustellen, die über das Internet verfügbar sind.

Sie können eine AWS Glue Kafka-Verbindung erstellen, um Ihre Verbindungsdaten zu verwalten. Weitere Informationen finden Sie unter the section called “Erstellen einer Verbindung für einen Kafka-Datenstrom”. Geben Sie in Ihrer AWS Glue-Job-Konfiguration *connectionName* als Zusätzliche Netzwerkverbindung an und geben Sie dann in Ihrem Methodenaufruf den `connectionName` Parameter *connectionName* an.

In bestimmten Fällen müssen Sie zusätzliche Voraussetzungen konfigurieren:

- Wenn Sie Amazon Managed Streaming für Apache Kafka mit IAM-Authentifizierung verwenden, benötigen Sie eine entsprechende IAM-Konfiguration.
- Wenn Sie Amazon Managed Streaming für Apache Kafka innerhalb einer Amazon VPC verwenden, benötigen Sie eine entsprechende Amazon-VPC-Konfiguration. Sie müssen eine AWS Glue-Verbindung erstellen, die Amazon VPC-Verbindungsinformationen bereitstellt. Sie benötigen Ihre Jobkonfiguration, um die AWS Glue-Verbindung als zusätzliche Netzwerkverbindung einzubeziehen.

Weitere Informationen zu den Voraussetzungen für Streaming-ETL-Aufträgen finden Sie unter the section called “Streaming-ETL-Aufträge”.

Beispiel: Aus Kafka-Streams lesen

Verwendet in Verbindung mit the section called “forEachBatch”.

Beispiel für die Kafka-Streaming-Quelle:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
```

```
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Beispiel: In Kafka-Streams schreiben

Beispiele für das Schreiben an Kafka:

Beispiel mit der `getSink` Methode:

```
data_frame_datasource0 =
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Beispiel mit der `write_dynamic_frame.from_options` Methode:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Referenz zur Kafka-Verbindungsoption

Verwenden Sie beim Lesen die folgenden Verbindungsoptionen mit`"connectionType": "kafka"`:

- `"bootstrap.servers"`(Erforderlich) Eine Liste von URLs Bootstrap-Servern, z. B. als`b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Diese Option muss im API-Aufruf angegeben oder in den Tabellenmetadaten im Data Catalog definiert werden.

- `"security.protocol"` (Erforderlich) Das Protokoll, das für die Kommunikation mit Brokern verwendet wird. Die möglichen Werte sind `"SSL"` oder `"PLAINTEXT"`.
- `"topicName"` (Erforderlich) Eine durch Kommas getrennte Liste von Themen, die abonniert werden sollen. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.
- `"assign"`: (Erforderlich) Eine JSON-Zeichenfolge, welche den spezifischen `TopicPartitions` zum Konsumieren angeben. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.

  Beispiel: '{"topicA":[0,1],"topicB":[2,4]}'
- `"subscribePattern"`: (Erforderlich) Eine Java-Regex-Zeichenfolge, die die Themenliste identifiziert, die abonniert werden soll. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.

  Beispiel: 'topic.*'
- `"classification"` (Erforderlich) Das von den Daten im Datensatz verwendete Dateiformat. Erforderlich, sofern nicht in Data Catalog angegeben.
- `"delimiter"` (Optional) Das verwendete Werttrennzeichen, wenn `classification` CSV ist. Der Standardwert ist „`,`„.
- `"startingOffsets"`: (Optional) Die Ausgangsposition im Kafka-Thema, aus dem Daten gelesen werden sollen. Die möglichen Werte sind `"earliest"` oder `"latest"`. Der Standardwert ist `"latest"`.
- `"startingTimestamp"`: (Optional, nur für AWS Glue Version 4.0 oder höher unterstützt) Der Zeitstempel des Datensatzes im Kafka-Thema, aus dem Daten gelesen werden sollen. Der mögliche Wert ist eine Zeitstempelzeichenfolge im UTC-Format im Muster `yyyy-mm-ddTHH:MM:SSZ` (wobei `Z` einen UTC-Zeitzonenversatz mit einem +/- darstellt. Beispiel: „2023-04-04T08:00:00-04:00“).

  Hinweis: In der Liste der Verbindungsoptionen des AWS Glue-Streaming-Skripts kann nur eine von 'StartingOffsets' oder 'StartingTimeStamp' vorhanden sein. Wenn Sie diese beiden Eigenschaften angeben, schlägt der Job fehl.
- `"endingOffsets"`: (Optional) Der Endpunkt, wenn eine Batchabfrage beendet wird. Die möglichen Werte sind entweder `"latest"` oder eine JSON-Zeichenfolge, die einen Offset für das Ende jeder `TopicPartition` angibt.

Für die JSON-Zeichenfolge lautet das Format `{"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}`. Der Wert `-1` als Offset steht für `"latest"`.

- `"pollTimeoutMs"`: (Optional) Das Timeout in Millisekunden, um Daten von Kafka in Spark-Auftragsausführungen abzufragen. Der Standardwert ist `600000`.
- `"numRetries"`: (Optional) Die Anzahl, wie oft erneute Versuche durchgeführt werden sollen, bevor Kafka-Offsets nicht abgerufen werden. Der Standardwert ist `3`.
- `"retryIntervalMs"`: (Optional) Die Wartezeit in Millisekunden, bevor Sie erneut versuchen, Kafka-Offsets abzurufen. Der Standardwert ist `10`.
- `"maxOffsetsPerTrigger"`: (Optional) Die Ratengrenze für die maximale Anzahl von Offsets, die pro Triggerintervall verarbeitet werden. Die angegebene Gesamtzahl der Offsets wird proportional auf `topicPartitions`von verschiedenen Volumes aufgeteilt. Der Standardwert ist null, was bedeutet, dass der Verbraucher alle Offsets bis zum bekannten letzten Offset liest.
- `"minPartitions"`: (Optional) Die gewünschte Mindestanzahl an Partitionen, die von Kafka gelesen werden sollen. Der Standardwert ist null, was bedeutet, dass die Anzahl der Spark-Partitionen gleich der Anzahl der Kafka-Partitionen ist.
- `"includeHeaders"`: (Optional) Gibt an, ob die Kafka-Header eingeschlossen werden sollen. Wenn die Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „glue_streaming_kafka_headers“ mit dem Typ `Array[Struct(key: String, value: String)]`. Der Standardwert ist „false“. Diese Option ist verfügbar in AWS Glue Version 3.0 oder höher.
- `"schema"`: (Erforderlich, wenn inferSchema auf „false“ festgelegt ist) Das Schema, das zur Verarbeitung der Nutzlast verwendet werden soll. Wenn die Klassifizierung `avro` ist, muss das bereitgestellte Schema im Avro-Schemaformat vorliegen. Wenn die Klassifizierung nicht `avro` ist, muss das bereitgestellte Schema im DDL-Schemaformat vorliegen.

  Im Folgenden finden Sie Beispiele für Schemata.

  Example in DDL schema format

  ```
  'column1' INT, 'column2' STRING , 'column3' FLOAT
  ```

  Example in Avro schema format

  ```
  {
  "type":"array",
  "items":
  ```

```
{
"type":"record",
"name":"test",
"fields":
[
  {
    "name":"_id",
    "type":"string"
  },
  {
    "name":"index",
    "type":
    [
      "int",
      "string",
      "float"
    ]
  }
]
}
}
```

- `"inferSchema"`: (Optional) Der Standardwert ist „false“. Wenn auf „true“ gesetzt, wird das Schema zur Laufzeit von der Nutzlast in `foreachbatch` erkannt.
- `"avroSchema"`: (Veraltet) Parameter, der verwendet wird, um ein Schema von Avro-Daten anzugeben, wenn das Avro-Format verwendet wird. Dieser Parameter ist jetzt veraltet. Verwenden Sie den Parameter `schema`.
- `"addRecordTimestamp"`: (Optional) Wenn diese Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die den Zeitpunkt angibt, zu dem der entsprechende Datensatz beim Thema eingegangen ist. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"emitConsumerLagMetrics"`: (Optional) Wenn die Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, den das Thema empfangen hat, und dem Zeitpunkt, zu dem er eingeht, ausgegeben AWS Glue zu CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.

Verwenden Sie beim Schreiben die folgenden Verbindungsoptionen mit`"connectionType": "kafka"`:

- `"connectionName"`(Erforderlich) Name der AWS Glue-Verbindung, die für die Verbindung mit dem Kafka-Cluster verwendet wird (ähnlich der Kafka-Quelle).
- `"topic"`(Erforderlich) Wenn eine Themenspalte existiert, wird ihr Wert als Thema verwendet, wenn die angegebene Zeile in Kafka geschrieben wird, sofern die Themenkonfigurationsoption nicht festgelegt ist. Das heißt, die `topic` Konfigurationsoption überschreibt die Themenspalte.
- `"partition"`(Optional) Wenn eine gültige Partitionsnummer angegeben ist, `partition` wird diese beim Senden des Datensatzes verwendet.

  Wenn keine Partition angegeben ist, aber eine vorhanden `key` ist, wird eine Partition anhand eines Hashs des Schlüssels ausgewählt.

  Falls `key` weder noch vorhanden `partition` ist, wird eine Partition auf der Grundlage von Sticky-Partitionierung ausgewählt. Diese Änderungen werden erst dann vorgenommen, wenn für die Partition mindestens batch.size-Byte erzeugt werden.
- `"key"`(Optional) Wird für die Partitionierung verwendet, wenn der Wert Null ist. `partition`
- `"classification"`(Optional) Das von den Daten im Datensatz verwendete Dateiformat. Wir unterstützen nur JSON, CSV und Avro.

  Mit dem Avro-Format können wir ein benutzerdefiniertes AvroSchema für die Serialisierung bereitstellen. Beachten Sie jedoch, dass dieses auch in der Quelle für die Deserialisierung bereitgestellt werden muss. Andernfalls verwendet es standardmäßig den Apache für die Serialisierung. AvroSchema

Darüber hinaus können Sie die Kafka-Senke nach Bedarf feinabstimmen, indem Sie die Konfigurationsparameter des Kafka-Producers aktualisieren. Beachten Sie, dass es keine Zulassungsliste für Verbindungsoptionen gibt, da alle Schlüssel-Wert-Paare unverändert auf der Senke gespeichert werden.

Es gibt jedoch eine kleine Liste von Optionen, die nicht wirksam werden. Weitere Informationen finden Sie unter Kafka-spezifische Konfigurationen.

Mit Kinesis-Verbindungen arbeiten

Sie können eine Kinesis-Verbindung zum Lesen und Schreiben in Amazon Kinesis Kinesis-Datenstreams verwenden, indem Sie Informationen verwenden, die in einer Datenkatalogtabelle gespeichert sind, oder indem Sie Informationen bereitstellen, um direkt auf den Datenstream zuzugreifen. Sie können Informationen aus Kinesis in einen Spark einlesen DataFrame und sie

dann in einen AWS Glue DynamicFrame umwandeln. Sie können in einem JSON-Format in Kinesis schreiben DynamicFrames . Wenn Sie direkt auf den Datenstrom zugreifen, verwenden Sie diese Optionen, um Informationen zum Zugriff auf den Datenstrom bereitzustellen.

Wenn Sie `getCatalogSource` oder `create_data_frame_from_catalog` verwenden, um Einträge aus einer Kinesis-Streamingquelle zu verbrauchen, enthält der Auftrag die Informationen zu Data-Catalog-Datenbank und Tabellennamen und kann diese verwenden, um einige grundlegende Parameter für das Lesen aus der Kinesis-Streaming-Quelle zu erhalten. Wenn Sie `getSource`, `getSourceWithFormat`, `createDataFrameFromOptions` oder `create_data_frame_from_options` verwenden, müssen Sie diese grundlegenden Parameter mithilfe der hier beschriebenen Verbindungsoptionen angeben.

Sie können die Verbindungsoptionen für Kinesis mit den folgenden Argumenten für die angegebenen Methoden in der `GlueContext`-Klasse angeben.

- Scala
  - `connectionOptions`: mit `getSource`, `createDataFrameFromOptions`, `getSink` verwenden
  - `additionalOptions`: mit `getCatalogSource`, `getCatalogSink` verwenden
  - `options`: mit `getSourceWithFormat`, `getSinkWithFormat` verwenden
- Python
  - `connection_options`: mit `create_data_frame_from_options`, `write_dynamic_frame_from_options` verwenden
  - `additional_options`: mit `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog` verwenden
  - `options`: mit `getSource`, `getSink` verwenden

Hinweise und Einschränkungen zu Streaming-ETL-Aufträgen finden Sie unter the section called “Hinweise zu und Einschränkungen für Streaming-ETL”.

Kinesis konfigurieren

Um in einem AWS Glue Spark-Job eine Verbindung zu einem Kinesis-Datenstream herzustellen, benötigen Sie einige Voraussetzungen:

- Beim Lesen muss der AWS Glue-Job über IAM-Berechtigungen auf Lesezugriffsebene für den Kinesis-Datenstream verfügen.

- Beim Schreiben muss der AWS Glue-Job über IAM-Berechtigungen auf Schreibzugriffsebene für den Kinesis-Datenstream verfügen.

In bestimmten Fällen müssen Sie zusätzliche Voraussetzungen konfigurieren:

- Wenn Ihr AWS Glue-Job mit zusätzlichen Netzwerkverbindungen konfiguriert ist (in der Regel, um eine Verbindung zu anderen Datensätzen herzustellen) und eine dieser Verbindungen Amazon VPC-Netzwerkoptionen bietet, leitet dies Ihren Job an, über Amazon VPC zu kommunizieren. In diesem Fall müssen Sie auch Ihren Kinesis-Datenstrom für die Kommunikation über Amazon VPC konfigurieren. Sie können dies tun, indem Sie einen Schnittstellen-VPC-Endpunkt zwischen Ihrer Amazon VPC und dem Kinesis-Datenstrom erstellen. Weitere Informationen finden Sie unter Verwenden von Kinesis Data Streams mit Schnittstellen-VPC-Endpunkten.
- Wenn Sie Amazon Kinesis Data Streams in einem anderen Konto angeben, müssen Sie die Rollen und Richtlinien einrichten, um den kontoübergreifenden Zugriff zu ermöglichen. Weitere Informationen finden Sie unter Beispiel: Aus einem Kinesis Stream in einem anderen Konto lesen.

Weitere Informationen zu den Voraussetzungen für Streaming-ETL-Aufträgen finden Sie unter the section called “Streaming-ETL-Aufträge”.

Aus Kinesis lesen

Beispiel: Lesen aus Kinesis-Streams

Verwendet in Verbindung mit the section called “forEachBatch”.

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

In Kinesis schreiben

Beispiel: In Kinesis-Streams schreiben

Verwendet in Verbindung mit the section called “forEachBatch”. Ihr DynamicFrame wird in einem JSON-Format in den Stream geschrieben. Wenn der Auftrag nach mehreren Versuchen nicht schreiben kann, schlägt er fehl. Standardmäßig wird jeder DynamicFrame Datensatz einzeln an den Kinesis-Stream gesendet. Sie können dieses Verhalten mithilfe von `aggregationEnabled` und zugehörigen Parametern konfigurieren.

Beispiel für das Schreiben in Amazon Kinesis von einem Streaming-Auftrag aus:

Python

```
glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite
    connection_type="kinesis",
    connection_options={
        "partitionKey": "part1",
        "streamARN": "arn:aws:kinesis:us-east-1:111122223333:stream/streamName",
    }
)
```

Scala

```
glueContext.getSinkWithFormat(
                connectionType="kinesis",
                options=JsonOptions("""{
                    "streamARN": "arn:aws:kinesis:us-
east-1:111122223333:stream/streamName",
                    "partitionKey": "part1"
                }"""),
           )
           .writeDynamicFrame(frameToWrite)
```

Kinesis-Verbindungsparameter

Bezeichnet Verbindungoptionen zu Amazon Kinesis Data Streams.

Verwenden Sie die folgenden Verbindungsoptionen für Kinesis-Streamingdatenquellen:

- "streamARN" (Erforderlich) Wird zum Lesen/Schreiben verwendet. Der ARN des Kinesis-Datenstroms.
- "classification" (Zum Lesen erforderlich) Wird zum Lesen verwendet. Das von den Daten im Datensatz verwendete Dateiformat. Erforderlich, sofern nicht in Data Catalog angegeben.
- "streamName" – (Optional) Wird zum Lesen verwendet. Der Name eines Kinesis-Datenstroms, aus dem gelesen wird. Wird mit endpointUrl verwendet.
- "endpointUrl" – (Optional) Wird zum Lesen verwendet. Standard: "https://kinesis.us-east-1.amazonaws.com“. Der AWS Endpunkt des Kinesis-Streams. Sie müssen dies nicht ändern, es sei denn, Sie stellen eine Verbindung zu einer bestimmten Region her.
- "partitionKey" – (Optional) Wird zum Schreiben verwendet. Der Kinesis-Partitionsschlüssel, der bei der Erstellung von Datensätzen verwendet wird.
- "delimiter" (Optional) Wird zum Lesen verwendet. Das verwendete Werttrennzeichen, wenn classification CSV ist. Der Standardwert ist „,„.
- "startingPosition": (Optional) Wird zum Lesen verwendet. Die Ausgangsposition im Kinesis Data Stream, von dem Daten gelesen werden sollen. Die möglichen Werte sind "latest", "trim_horizon", "earliest" oder eine Zeitstempelzeichenfolge im UTC-Format im Muster yyyy-mm-ddTHH:MM:SSZ (wobei Z einen UTC-Zeitzonenversatz mit einem +/- darstellt. Zum Beispiel „2023-04-04T08:00:00-04:00“). Der Standardwert ist "latest". Hinweis: Die Timestamp-Zeichenfolge im UTC-Format für "startingPosition" wird nur für AWS Glue Version 4.0 oder höher unterstützt.
- "failOnDataLoss": (Optional) Lassen Sie den Auftrag fehlschlagen, wenn ein aktiver Shard fehlt oder abgelaufen ist. Der Standardwert ist "false".
- "awsSTSRoleARN": (Optional) Wird zum Schreiben/Lesen verwendet. Der Amazon-Ressourcenname (ARN) der Rolle, die mithilfe von AWS Security Token Service (AWS STS) übernommen werden soll. Diese Rolle muss über Berechtigungen zum Beschreiben oder Lesen von Datensatzoperationen für den Kinesis-Datenstrom verfügen. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit "awsSTSSessionName".
- "awsSTSSessionName": (Optional) Wird zum Schreiben/Lesen verwendet. Ein Bezeichner für die Sitzung, die die Rolle unter Verwendung von AWS STSübernimmt. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit "awsSTSRoleARN".
- "awsSTSEndpoint": (Optional) Der AWS STS Endpunkt, der verwendet werden soll, wenn eine Verbindung zu Kinesis mit einer angenommenen Rolle hergestellt wird. Dies ermöglicht

die Verwendung des regionalen AWS STS Endpunkts in einer VPC, was mit dem globalen Standardendpunkt nicht möglich ist.

- `"maxFetchTimeInMs"`: (Optional) Wird zum Lesen verwendet. Die maximale Zeit, die der Job Executor benötigt, um Datensätze für den aktuellen Batch aus dem Kinesis-Datenstream zu lesen, angegeben in Millisekunden (ms). Innerhalb dieser Zeit können mehrere `GetRecords` API-Aufrufe getätigt werden. Der Standardwert ist `1000`.
- `"maxFetchRecordsPerShard"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl von Datensätzen, die pro Shard im Kinesis-Datenstrom pro Mikrobatch abgerufen werden können. Hinweis: Der Client kann dieses Limit überschreiten, wenn der Streaming-Job bereits zusätzliche Datensätze von Kinesis gelesen hat (im selben Get-Records-Aufruf). Wenn es streng sein `maxFetchRecordsPerShard` muss, muss es ein Vielfaches von sein. `maxRecordPerRead` Der Standardwert ist `100000`.
- `"maxRecordPerRead"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl von Datensätzen, die bei jeder `getRecords`-Operation aus dem Kinesis-Datenstrom abgerufen werden sollen. Der Standardwert ist `10000`.
- `"addIdleTimeBetweenReads"`: (Optional) Wird zum Lesen verwendet. Fügt eine Zeitverzögerung zwischen zwei aufeinanderfolgenden `getRecords`-Operationen ein. Der Standardwert ist `"False"`. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.
- `"idleTimeBetweenReadsInMs"`: (Optional) Wird zum Lesen verwendet. Die minimale Zeitverzögerung zwischen zwei aufeinanderfolgenden `getRecords`-Operationen, angegeben in Millisekunden (ms). Der Standardwert ist `1000`. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.
- `"describeShardInterval"`: (Optional) Wird zum Lesen verwendet. Das minimale Zeitintervall zwischen zwei `ListShards`-API-Aufrufen für die Erwägung eines erneuten Shardings durch Ihr Skript. Weitere Informationen finden Sie unter Strategies for Resharding (Strategien für das Resharding) im Entwicklerhandbuch für Amazon Kinesis Data Streams. Der Standardwert ist `1s`.
- `"numRetries"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl erneuter Versuche für API-Aufrufe von Kinesis Data Streams. Der Standardwert ist `3`.
- `"retryIntervalMs"`: (Optional) Wird zum Lesen verwendet. Die Abkühlzeit (angegeben in ms) vor dem erneuten Versuch des API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `1000`.
- `"maxRetryIntervalMs"`: (Optional) Wird zum Lesen verwendet. Die maximale Abkühlzeit (angegeben in ms) zwischen zwei wiederholten Versuchen eines API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `10000`.

- `"avoidEmptyBatches"`: (Optional) Wird zum Lesen verwendet. Vermeidet das Erstellen eines leeren Mikrobatchauftrags, indem vor dem Start des Batches im Kinesis Data Stream nach ungelesenen Daten gesucht wird. Der Standardwert ist `"False"`.
- `"schema"`: (Erforderlich, wenn inferSchema auf „false“ festgelegt ist) Wird zum Lesen verwendet. Das zu verwendende Schema für die Verarbeitung der Nutzlast. Wenn die Klassifizierung `avro` ist, muss das bereitgestellte Schema im Avro-Schemaformat vorliegen. Wenn die Klassifizierung nicht `avro` ist, muss das bereitgestellte Schema im DDL-Schemaformat vorliegen.

  Im Folgenden finden Sie Beispiele für Schemata.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
    {
      "type":"record",
      "name":"test",
      "fields":
      [
        {
          "name":"_id",
          "type":"string"
        },
        {
          "name":"index",
          "type":
          [
            "int",
            "string",
            "float"
          ]
        }
      ]
    }
  }
  ```

- `"inferSchema"`: (Optional) Wird zum Lesen verwendet. Der Standardwert von "false". Wenn auf „true“ gesetzt, wird das Schema zur Laufzeit von der Nutzlast in `foreachbatch` erkannt.
- `"avroSchema"`: (Veraltet) Wird zum Lesen verwendet. Der Parameter, der verwendet wird, um ein Schema von Avro-Daten anzugeben, wenn das Avro-Format verwendet wird. Dieser Parameter ist jetzt veraltet. Verwenden Sie den Parameter `schema`.
- `"addRecordTimestamp"`: (Optional) Wird zum Lesen verwendet. Wenn diese Option auf 'true' gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die die Uhrzeit angibt, zu der der entsprechende Datensatz mit dem Stream empfangen wurde. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"emitConsumerLagMetrics"`: (Optional) Wird zum Lesen verwendet. Wenn die Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, der vom Stream empfangen wurde, und dem Zeitpunkt, zu dem er eingeht, ausgegeben AWS Glue zu CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"fanoutConsumerARN"`: (Optional) Wird zum Lesen verwendet. Der ARN eines Kinesis-Stream-Verbrauchers für den in `streamARN` angegebenen Stream. Wird verwendet, um den erweiterten Fan-Out-Modus für Ihre Kinesis-Verbindung zu aktivieren. Weitere Informationen zur Nutzung eines Kinesis-Streams mit erweitertem Fan-Out finden Sie unter the section called “Verwendung von erweitertem Fan-Out in Kinesis-Streaming-Aufträgen”.
- `"recordMaxBufferedTime"` – (Optional) Wird zum Schreiben verwendet. Standard: 1.000 (ms). Maximale Dauer, für die ein Datensatz gepuffert wird, während er darauf wartet, geschrieben zu werden.
- `"aggregationEnabled"` – (Optional) Wird zum Schreiben verwendet. Standard: true Gibt an, ob Datensätze aggregiert werden sollen, bevor sie an Kinesis gesendet werden.
- `"aggregationMaxSize"` – (Optional) Wird zum Schreiben verwendet. Standard: 51.200 (Byte). Wenn ein Datensatz dieses Limit übersteigt, wird der Aggregator umgangen. Hinweis Kinesis erzwingt ein Limit von 50 KB für die Größe eines Datensatzes. Wenn Sie diesen Wert auf mehr als 50 KB festlegen, werden übergroße Datensätze von Kinesis zurückgewiesen.
- `"aggregationMaxCount"` – (Optional) Wird zum Schreiben verwendet. Standard: 4.294.967.295. Maximale Anzahl von Elementen, die in einen aggregierten Datensatz gepackt werden sollen.
- `"producerRateLimit"` – (Optional) Wird zum Schreiben verwendet. Standard: 150 (%). Beschränkt den Durchsatz pro Shard, der von einem einzelnen Produzenten (z. B. Ihrem Auftrag) gesendet wird, als prozentualen Wert des Backend-Limits.

- `"collectionMaxCount"` – (Optional) Wird zum Schreiben verwendet. Standard: 500. Maximale Anzahl von Elementen, die in eine PutRecords Anfrage gepackt werden sollen.
- `"collectionMaxSize"` – (Optional) Wird zum Schreiben verwendet. Standard: 5.242.880 (Byte). Maximale Datenmenge, die mit einer PutRecords Anfrage gesendet werden kann.

AWS Glue Automatische Streaming-Skalierung

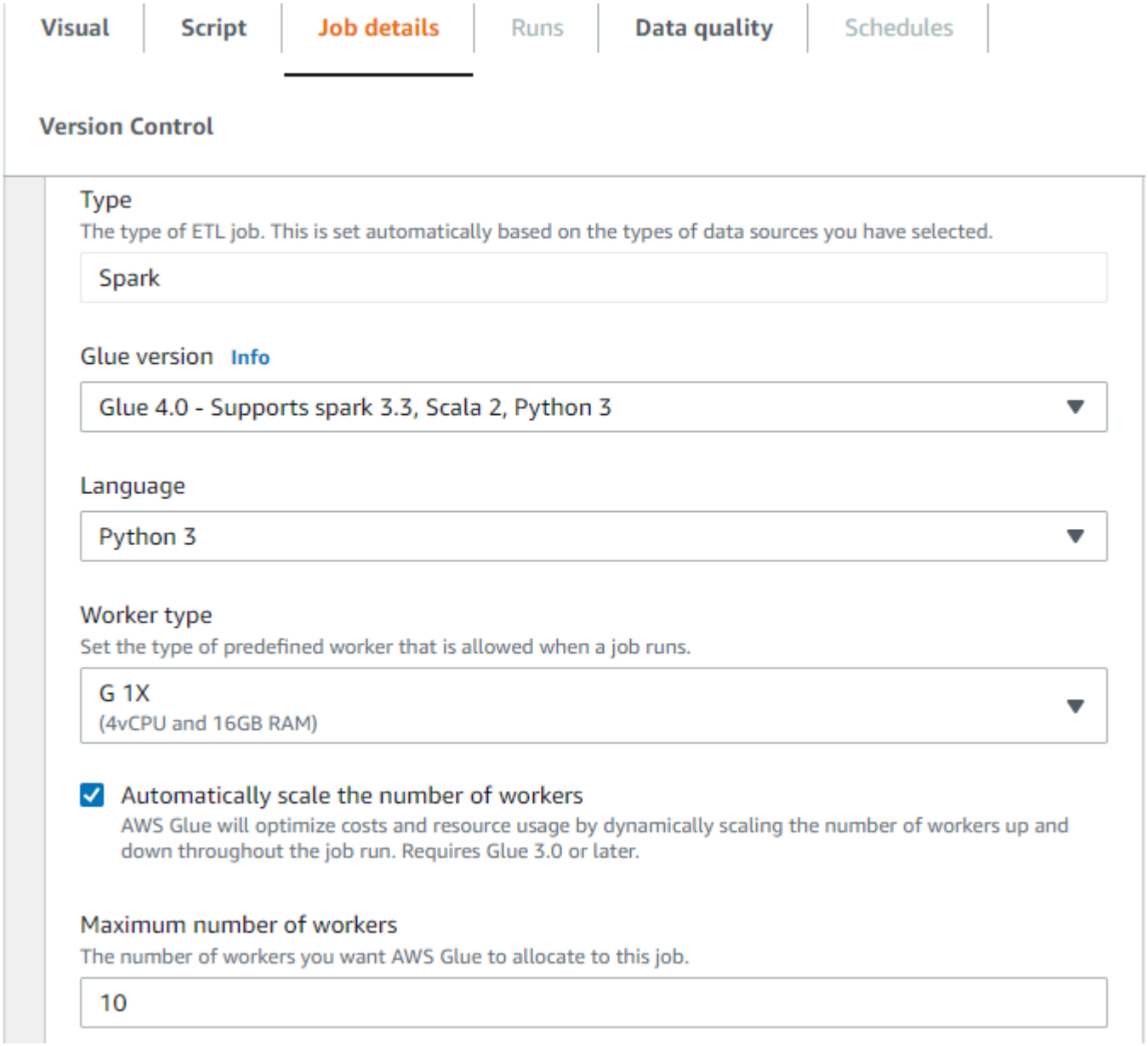
AWS Glue Streaming-ETL-Jobs verbrauchen kontinuierlich Daten aus Streaming-Quellen, bereinigen und transformieren die Daten während der Übertragung und stellen sie für Analysen zur Verfügung. Durch die Überwachung jeder Phase der Auftragsausführung kann AWS Glue Autoscaling Worker ausschalten, wenn sie inaktiv sind, oder Worker hinzufügen, wenn zusätzliche Parallelverarbeitung möglich ist.

Die folgenden Abschnitte enthalten Informationen zur automatischen AWS Glue Streaming-Skalierung

Aktivieren von Auto Scaling in AWS Glue Studio

Wählen Sie in AWS Glue Studio auf der Registerkarte Jobdetails den Typ als Spark oder Spark Streaming und die Glue-Version als **`Glue 3.0`** oder aus**`Glue 4.0`**. Dann erscheint unter Worker-Typ ein Kontrollfeld.

- Wählen Sie die Option Automatisches Skalieren der Worker-Anzahl aus.
- Legen Sie die Maximale Worker-Anzahl fest, um die maximale Anzahl von Workern zu definieren, die für die Auftragsausführung ausgegeben werden können.



Auto Scaling mit der AWS CLI oder dem SDK aktivieren

Um Auto Scaling über die AWS CLI für Ihren Joblauf zu aktivieren, führen Sie `start-job-run` ihn mit der folgenden Konfiguration aus:

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
```

```
    "WorkerType": "G.2X", // G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X,
 and R.8X are supported for Auto Scaling Jobs
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Wenn die ETL-Auftragsausführung abgeschlossen ist, können Sie auch `get-job-run` aufrufen, um die tatsächliche Ressourcennutzung der Ausführung in DPU-Sekunden zu prüfen. Hinweis: Das neue Feld DPUSecondswird nur für Ihre Batch-Jobs auf AWS Glue Version 3.0 oder höher angezeigt, die mit Auto Scaling aktiviert sind. Dieses Feld wird für Streaming-Aufträge nicht unterstützt.

```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Sie können Auftragsausführungen mit Auto Scaling auch über das AWS Glue -SDK konfigurieren. Die Konfiguration ist dieselbe.

Funktionsweise

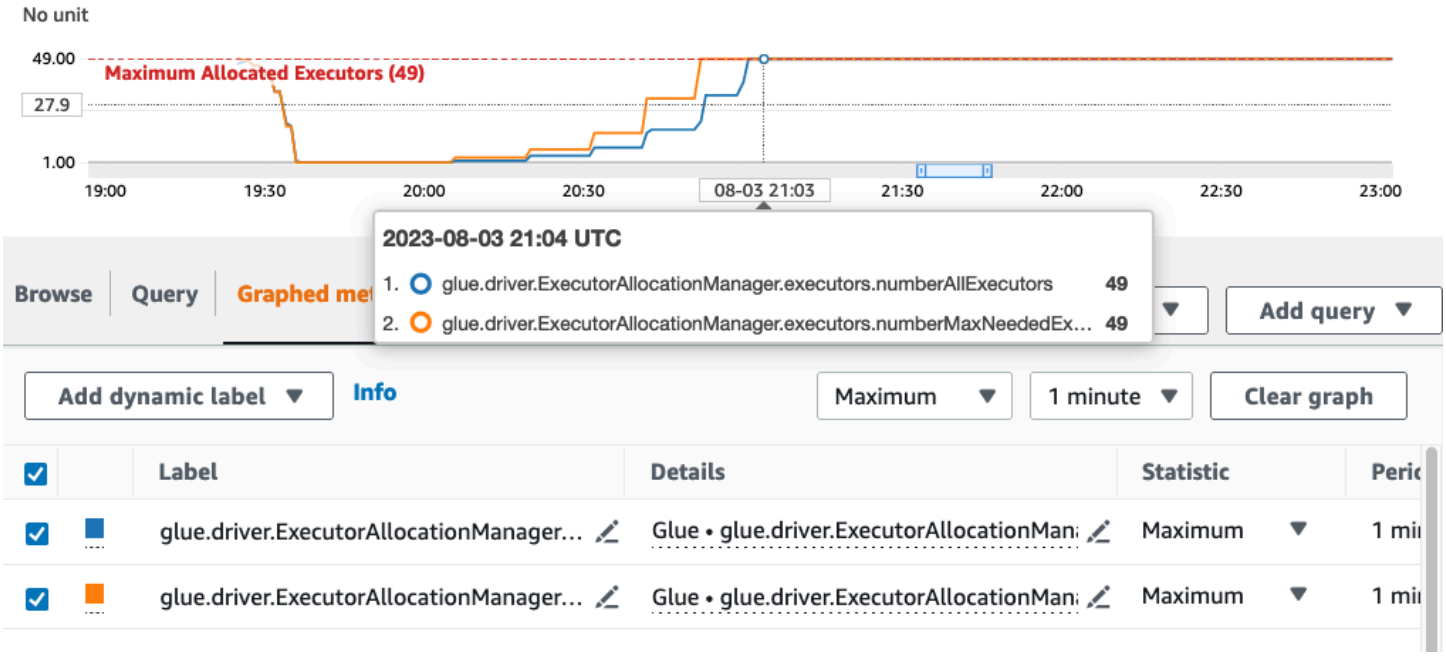
Skalieren über Micro-Batches hinweg

Anhand des folgenden Beispiels wird beschrieben, wie Auto Scaling funktioniert.

- Sie haben einen AWS Glue Job, der mit 50 beginnt DPUs.
- Auto Scaling ist aktiviert.

In diesem Beispiel wird AWS Glue anhand der Metrik batchProcessingTime InMs „„für einige Mikrobatches ermittelt, ob Ihre Aufträge innerhalb der von Ihnen festgelegten Fenstergröße abgeschlossen werden. Wenn Ihre Aufträge schneller abgeschlossen werden, und je nachdem, wie schnell sie abgeschlossen werden, kann es sein, dass AWS Glue herunterskaliert wird. Diese Metrik, dargestellt mit“ numberAllExecutors „, kann überwacht werden, Amazon CloudWatch um zu sehen, wie Autoscaling funktioniert.

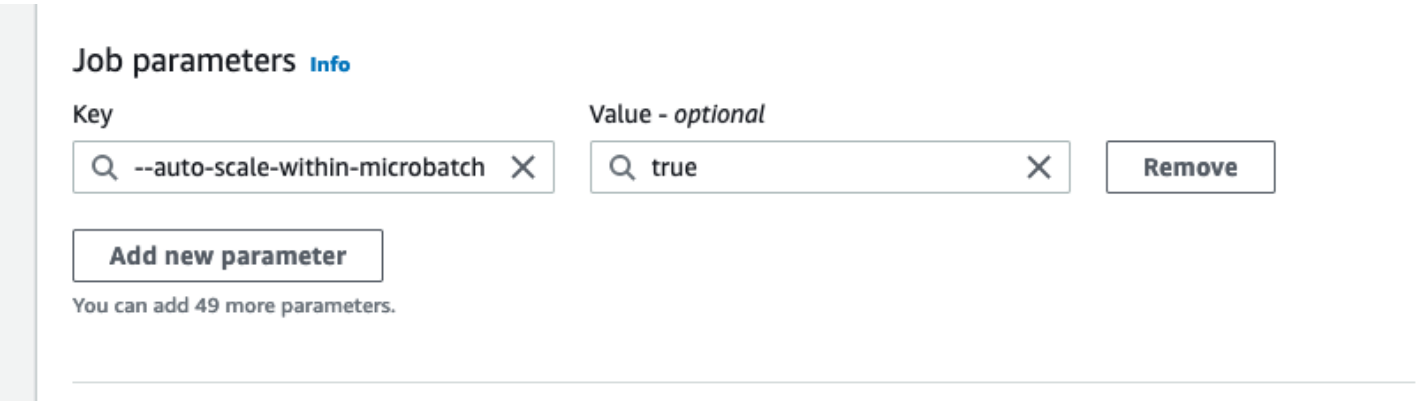
Die Anzahl der Executors skaliert exponentiell nach oben oder unten, nachdem jeder Micro-Batch abgeschlossen ist. Wie Sie dem Amazon CloudWatch Monitoring-Protokoll entnehmen können, AWS Glue wird die Anzahl der benötigten Executors (orange Linie) überprüft und die Executoren (blaue Linie) automatisch entsprechend skaliert.



Sobald die Anzahl der Executoren AWS Glue herunterskaliert wird und festgestellt wird, dass das Datenvolumen zunimmt, wodurch sich die Verarbeitungszeit für Mikro-Batches verlängert, AWS Glue wird die Skalierung auf 50 DPUs erhöht, was der angegebenen Obergrenze entspricht.

Skalieren innerhalb von Micro-Batches

Im obigen Beispiel überwacht das System einige abgeschlossene Micro-Batches, um zu entscheiden, ob hoch- oder herunterzuskalieren ist. Längere Zeitfenster erfordern Autoscaling, um innerhalb des Mikrobatches schneller reagieren zu können, anstatt auf einige Mikrobatches zu warten. Für diese Fälle können Sie eine zusätzliche Konfiguration `--auto-scaIe-within-microbatch` zu `true` verwenden. Sie können dies AWS Glue Studio wie unten gezeigt zu den AWS Glue Auftragseigenschaften hinzufügen.



Wartungsfenster für AWS Glue Streaming

AWS Glue führt regelmäßig Wartungsaktivitäten durch. Während dieser Wartungsfenster müssen AWS Glue Sie Ihre Streaming-Jobs neu starten. Sie können steuern, wann die Jobs neu gestartet werden, indem Sie Wartungsfenster angeben. In diesem Abschnitt erläutern wir, wo Sie das Wartungsfenster einrichten können und welche Verhaltensweisen Sie dabei berücksichtigen sollten.

Themen

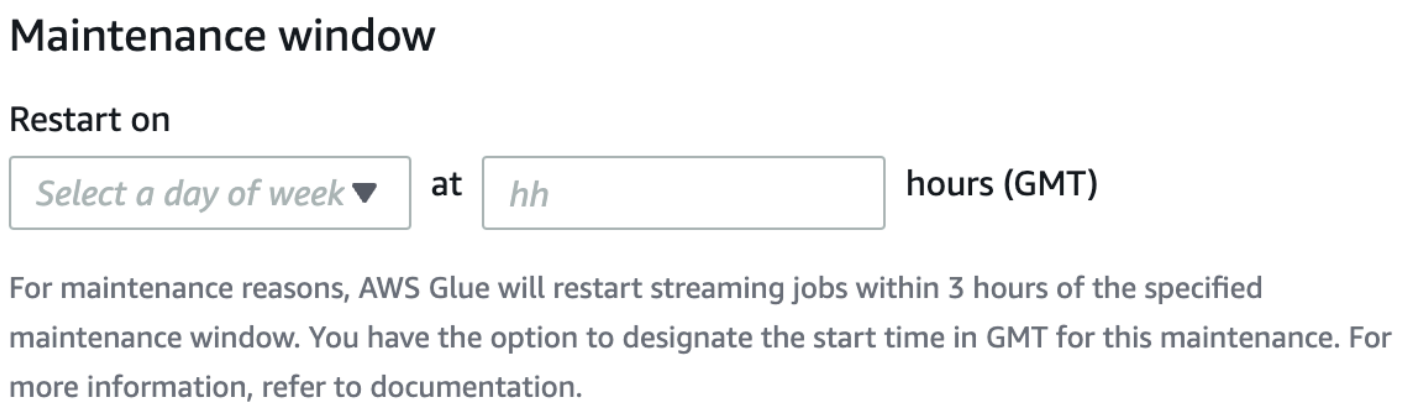
- Ein Wartungsfenster einrichten
- Verhalten des Wartungsfensters
- Überwachung von Job
- Umgang mit Datenverlust

Ein Wartungsfenster einrichten

Sie können ein Wartungsfenster mit AWS Glue Studio oder einrichten APIs.

Ein Wartungsfenster in AWS Glue Studio einrichten

Sie können auf der Seite mit den Jobdetails Ihres AWS Glue Streaming-Jobs ein Wartungsfenster angeben. Sie können den Tag und die Uhrzeit in GMT angeben. AWS Glue wird Ihren Job innerhalb des angegebenen Zeitfensters neu starten.



Ein Wartungsfenster in der API einrichten

Sie können das Wartungsfenster alternativ in der Create Job API einrichten. Hier ist ein Beispiel für die Konfiguration eines Wartungsfensters über die API.

```
aws glue create-job –name jobName –role roleArnForTheJob –command
 Name=gluestreaming,ScriptLocation=s3-path-to-the-script --maintenance-window="Sun:10"
```

Ein Beispielbefehl lautet wie folgt:

```
aws glue create-job –name testMaintenance –role arn:aws:iam::012345678901:role/
Glue_DefaultRole –command Name=gluestreaming,ScriptLocation=s3://glue-example-test/
example.py –maintenance-window="Sun:10
```

Verhalten des Wartungsfensters

AWS Glue durchläuft eine Reihe von Schritten, um zu entscheiden, wann ein Job neu gestartet werden soll:

1. Wenn ein neuer Streaming-Job initiiert wird, wird AWS Glue zunächst geprüft, ob mit der Jobausführung ein Timeout verbunden ist. Ein Timeout ermöglicht es Ihnen, die Endzeit des Jobs zu konfigurieren. Wenn das Timeout weniger als 7 Tage beträgt, wird der Job nicht neu gestartet.
2. Wenn das Timeout länger als 7 Tage ist, wird AWS Glue geprüft, ob das Wartungsfenster für den Job konfiguriert ist. Ist dies der Fall, wird dieses Fenster ausgewählt und das Fenster wird der Auftragsausführung zugewiesen. AWS Glue startet den Job innerhalb von 3 Stunden nach Ablauf des angegebenen Wartungsfensters neu. Wenn Sie beispielsweise das Wartungsfenster für

Montag um 10:00 Uhr GMT einrichten, werden Ihre Jobs zwischen 10:00 Uhr GMT und 13:00 Uhr GMT neu gestartet.

3. Wenn das Wartungsfenster nicht konfiguriert ist, wird die Neustartzeit AWS Glue automatisch auf 7 Tage nach der Initiierung der Auftragsausführung festgelegt. Wenn Sie Ihren Job beispielsweise am 01.07.2024 12:00 Uhr GMT initiiert haben und keine Wartungsfenster angegeben haben, wird Ihr Job so eingestellt, dass er am 08.07.2024 um 12:00 Uhr GMT neu gestartet wird.

> **Note**
>
> Wenn Sie bereits Streaming-Jobs ausführen, wirkt sich diese Änderung ab dem 1. Juli 2024 auf Sie aus. Sie haben bis zum 30. Juni Zeit, Ihre Wartungsfenster zu konfigurieren. Nach dem 1. Juli werden alle Streaming-Jobs, die Sie starten, gemäß dieser Dokumentation neu gestartet. Wenn Sie zusätzliche Unterstützung benötigen, können Sie sich an den AWS Support wenden.

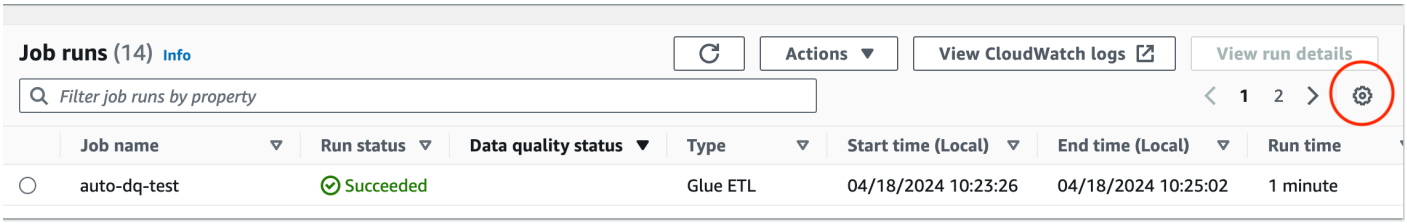
4. Manchmal kann der Job AWS Glue möglicherweise nicht neu gestartet werden, insbesondere wenn der laufende Mikrobatch nicht verarbeitet wird. In diesen Fällen wird der Job nicht unterbrochen. In diesen Fällen AWS Glue wird der Job nach 14 Tagen neu gestartet. In diesem Fall wird das Wartungsfenster nicht eingehalten.

Überwachung von Job

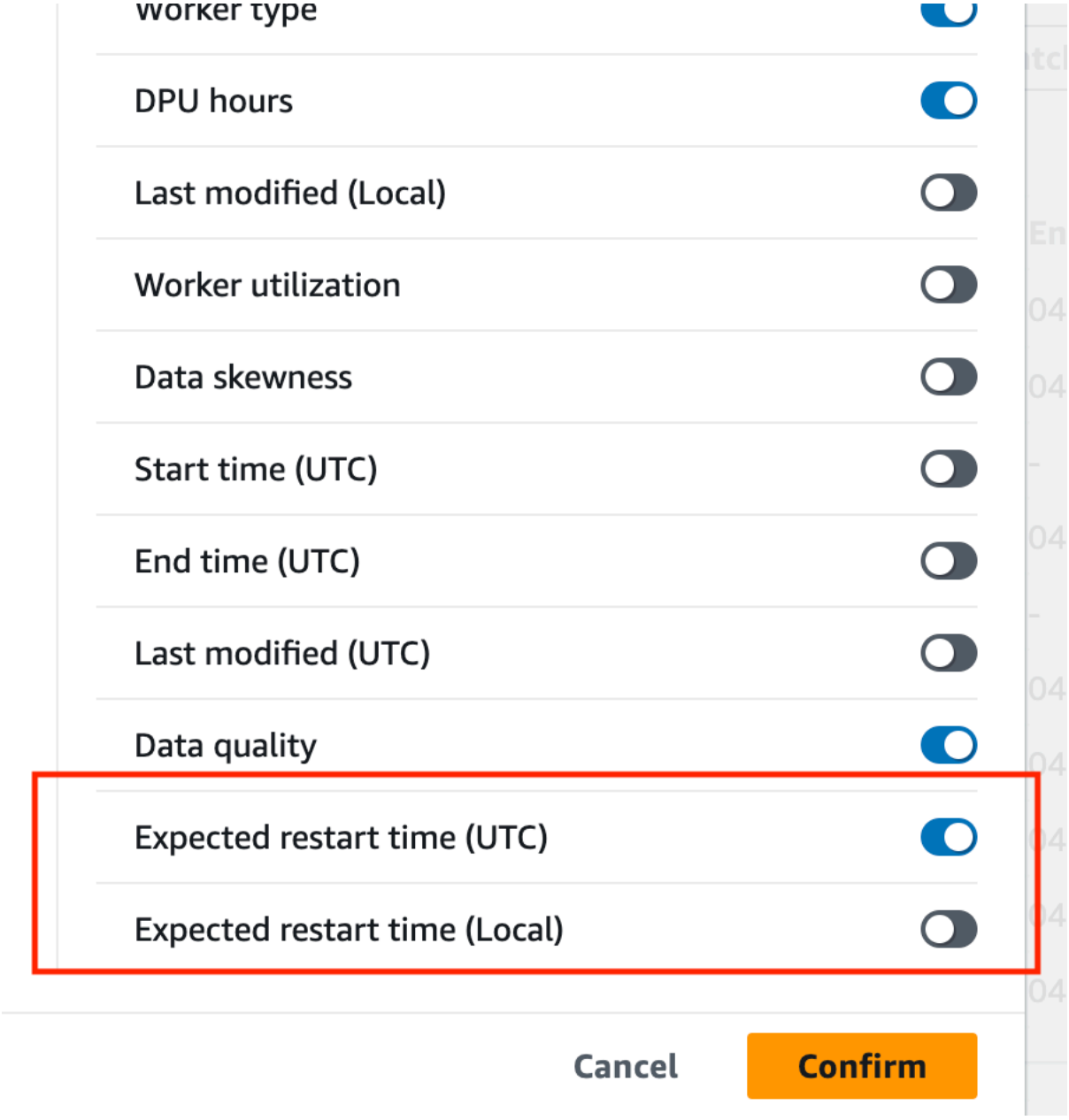
Sie können die Jobs auf der AWS Glue Studio-Monitoring-Seite überwachen.

Um den voraussichtlichen Zeitpunkt des nächsten Neustarts von Streaming-Jobs zu sehen, zeigen Sie die Spalte in der Tabelle Auftragsausführungen auf der Seite Überwachung an.

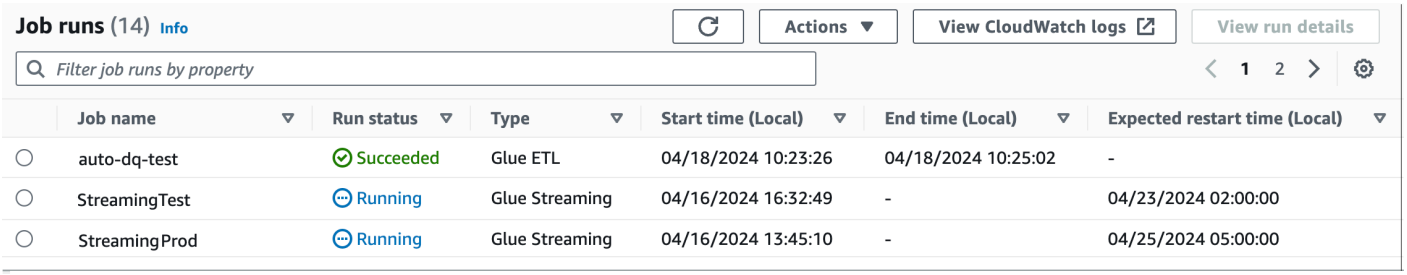
1. Klicken Sie oben rechts in der Tabelle auf das Zahnradsymbol.



2. Scrollen Sie nach unten und aktivieren Sie die Spalte Erwartete Neustartzeit. Es sind sowohl UTC- als auch Lokalzeitoptionen verfügbar.



3. Anschließend können Sie die Spalten in der Tabelle anzeigen.



Der ursprüngliche Job wird den Status „ABGELAUFEN“ haben und die neue Auftragsinstanz den Status „LÄUFT“. Die neue Auftragsausführung, die neu gestartet wurde, hat eine Job-Run-ID als Verkettung aus der ID der ersten Auftragsausführung plus dem Präfix „restart_“, das die Anzahl der Neustarts darstellt. Wenn die ID der ersten Auftragsausführung beispielsweise lautet`jr_1234`, hat die neu gestartete Auftragsausführung die ID für den ersten Neustart. `jr1234_restart_1` Der zweite Neustart ist `jr1234_restart_2` für den zweiten Neustart und so weiter vorgesehen.

Ihr Wiederholungsversuch wird durch die Neustarts nicht beeinträchtigt. Wenn ein Lauf fehlschlägt und aufgrund eines automatischen Wiederholungsversuchs ein neuer Lauf gestartet wird, beginnt der Neustartzähler wieder bei 1. Schlägt ein Lauf beispielsweise bei fehl`jr_1234_attempt_3_restart_5`, wird bei einem automatischen Wiederholungsversuch ein neuer Lauf mit der ID: gestartet, `jr_id1_attempt_4` und wenn dieser Versuch nach 7 Tagen erneut gestartet wird, lautet die neue Lauf-ID. `jr_id1_attempt_4_restart_1`

Umgang mit Datenverlust

Bei Wartungsneustarts folgt AWS Glue Streaming einem Prozess, der die Datenintegrität und Konsistenz zwischen der vorherigen Auftragsausführung und der neu gestarteten Auftragsausführung sicherstellt. Beachten Sie, dass AWS Glue dies nicht die Datenintegrität und Konsistenz zwischen Auftragsneustarts garantiert. Wir empfehlen daher, Überlegungen zur Architektur zu beachten, um doppelte Daten innerhalb von Streaming-Aufträgen zu behandeln.

1. Erkennung von Wartungsneustartbedingungen: AWS Glue Streaming überwacht Bedingungen, die angeben, wann ein Wartungsneustart ausgelöst werden sollte, z. B. wenn nach 7 Tagen ein Wartungsfenster erreicht wird oder nach 14 Tagen ein harter Neustart erforderlich ist.
2. Aufrufen eines ordnungsgemäßen Abbruchs: Wenn die Bedingungen für den Neustart der Wartung erfüllt sind, leitet AWS Glue Streaming einen ordnungsgemäßen Beendungsprozess für den aktuell ausgeführten Job ein. Dieser Prozess umfasst die folgenden Schritte:
   a. Stoppen der Aufnahme neuer Daten: Der Streaming-Job verwendet keine neuen Daten mehr aus den Eingabequellen (z. B. Kafka-Themen, Kinesis-Streams oder Dateien).
   b. Verarbeitung ausstehender Daten: Der Job verarbeitet weiterhin alle Daten, die sich bereits in seinen internen Puffern oder Warteschlangen befinden.
   c. Offsets und Checkpoints festschreiben: Der Job überträgt die neuesten Offsets oder Checkpoints an externe Systeme (z. B. Kafka, Kinesis oder Amazon S3), um sicherzustellen, dass der neu gestartete Job dort weitermachen kann, wo der vorherige Job aufgehört hat.
3. Job neu starten: Nachdem der Vorgang zur ordnungsgemäßen Beendigung abgeschlossen ist, startet Streaming den Job mit dem beibehaltenen Status und den Checkpoints neu. AWS Glue

Der neu gestartete Job nimmt die Verarbeitung ab dem letzten festgeschriebenen Offset oder Checkpoint auf und stellt so sicher, dass keine Daten verloren gehen oder dupliziert werden.

4. Fortsetzung der Datenverarbeitung: Der neu gestartete Job setzt die Datenverarbeitung an dem Punkt fort, an dem der vorherige Job aufgehört hat. Er setzt die Aufnahme neuer Daten aus den Eingabequellen fort, beginnend mit dem letzten festgeschriebenen Offset oder Checkpoint, und verarbeitet die Daten gemäß der definierten ETL-Logik.

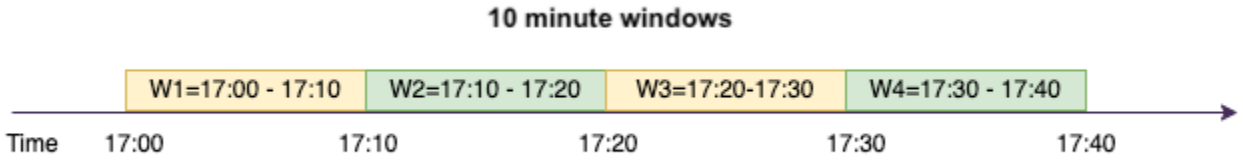
Advanced AWS Glue Streaming-Konzepte

In modernen datengesteuerten Anwendungen nimmt die Bedeutung von Daten mit der Zeit ab und ihr Wert geht von prädiktiv zu reaktiv über. Kunden wollen daher Daten in Echtzeit verarbeiten, um schneller Entscheidungen treffen zu können. Bei Echtzeit-Datenerfassungen, z. B. von IoT-Sensoren, können Daten ungeordnet ankommen oder aufgrund von Netzwerklatenz und anderen quellenbedingten Fehlern während der Erfassung Verzögerungen bei der Verarbeitung aufweisen. Im Rahmen der AWS Glue plattform, AWS Glue Streaming baut auf diesen Funktionen auf und bietet skalierbares, serverloses Streaming-ETL, das auf dem strukturierten Streaming von Apache Spark basiert und Benutzern die Datenverarbeitung in Echtzeit ermöglicht.

In diesem Thema werden wir erweiterte Streaming-Konzepte und -Funktionen von untersuchen AWS Glue Streaming.

Zeitliche Überlegungen bei der Verarbeitung von Streams

Bei der Verarbeitung von Streams gibt es vier Konzepte zum Thema Zeit:



- Event-time – Die Zeit, zu der das Ereignis eingetreten ist. In den meisten Fällen ist dieses Feld in der Quelle in die Ereignisdaten selbst eingebettet.
- E vent-time-window — Der Zeitrahmen zwischen zwei Ereigniszeiten. Wie im obigen Diagramm dargestellt, liegt W1 event-time-window zwischen 17:00 und 17:10. Bei jedem Ereignis event-time-window handelt es sich um eine Gruppierung mehrerer Ereignisse.

- Trigger-time – Die Auslösezeit steuert, wie oft die Verarbeitung von Daten und die Aktualisierung der Ergebnisse erfolgen. Dies ist der Zeitpunkt, an dem die Verarbeitung von Micro-Batches beginnt.
- Ingestion-time – Der Zeitpunkt, zu dem die Stream-Daten im Streaming-Service erfasst wurden. Wenn die Ereigniszeit nicht in das Ereignis selbst eingebettet ist, kann diese Zeit in einigen Fällen für das Windowing verwendet werden.

Windowing

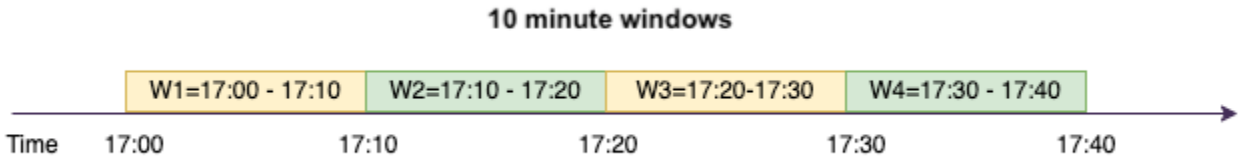
Windowing ist eine Technik, bei der Sie mehrere Ereignisse nach Gruppen gruppieren und aggregieren. event-time-window In den folgenden Beispielen werden wir uns mit den Vorteilen von Windowing befassen und herausfinden, wann Sie es verwenden sollten.

Je nach geschäftlichem Anwendungsfall gibt es drei Arten von Zeitfenstern, die von Spark unterstützt werden.

- Tumbling-Window — eine Reihe von sich nicht überlappenden, festen Größen, event-time-windows über die Sie aggregieren.
- Gleitendes Fenster – ähnlich wie bei den taumelnden Fenstern, wobei die Fenster eine „feste Größe“ haben, aber sie können sich überlappen oder verschieben, solange die Laufzeit der Verschiebung kleiner ist als die Laufzeit des Fensters selbst.
- Sitzungsfenster – startet mit einem Eingabedaten-Ereignis und erweitert sich selbst, solange es innerhalb einer Lücke oder Inaktivitätsdauer Eingaben erhält. Ein Sitzungsfenster kann eine statische oder dynamische Größe der Fensterlaufzeit haben, abhängig von den Eingaben.

Rollierendes Fenster

Bei einem Taumelfenster handelt es sich um eine Reihe von sich nicht überlappenden festen Größen, über die Sie aggregieren. event-time-windows Machen wir uns das anhand eines Beispiels aus der Praxis klar.



Das Unternehmen ABC Auto möchte eine Marketingkampagne für eine neue Sportwagenmarke durchführen. Sie wollen eine Stadt auswählen, in der sie die größten Fans von Sportwagen vorfinden. Um dieses Ziel zu erreichen, zeigen sie auf ihrer Website einen kurzen 15-sekündigen Werbespot, der das Auto vorstellt. Alle „Klicks“ und die entsprechende“ Stadt „werden aufgezeichnet und gestreamt. Amazon Kinesis Data Streams Wir möchten die Anzahl der Klicks in einem 10-minütigen Zeitfenster zählen und sie nach Städten gruppieren, um zu sehen, welche Stadt die höchste Nachfrage aufweist. Im Folgenden sehen Sie die Ausgabe dieser Aggregation.

window_start_time	window_end_time	city	total_clicks
2023-07-10 17:00:00	2023-07-10 17:10:00	Dallas	75
2023-07-10 17:00:00	2023-07-10 17:10:00	Chicago	10
2023-07-10 17:20:00	2023-07-10 17:30:00	Dallas	20
2023-07-10 17:20:00	2023-07-10 17:30:00	Chicago	50

Wie oben erläutert, unterscheiden sich diese von den event-time-windows Triggerzeitintervallen. Selbst wenn Sie beispielsweise jede Minute einen Trigger auslösen, werden die Ausgabeergebnisse nur 10 Minuten nicht überlappende Aggregationsfenster zeigen. Zur Optimierung ist es besser, das Triggerintervall an den auszurichten. event-time-window

In der obigen Tabelle verzeichnete Dallas 75 Klicks im Fenster 17.00–17.10, während Chicago 10 Klicks hatte. Außerdem gibt es für keine Stadt Daten für das Fenster 17.10–17.20, so dass dieses Fenster ausgelassen wird.

Jetzt können Sie diese Daten in der nachgelagerten Analyseanwendung weiter analysieren, um die attraktivste Stadt für die Durchführung der Marketingkampagne zu ermitteln.

Verwenden Sie taumelnde Fenster in AWS Glue

1. Erstellen Sie eine Amazon Kinesis Data Streams DataFrame und lesen Sie daraus. Beispiel:

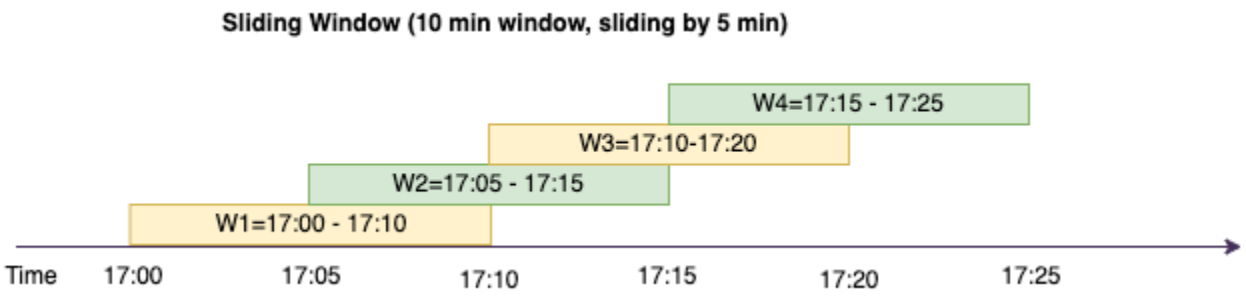
```
parsed_df = kinesis_raw_df \
                .selectExpr('CAST(data AS STRING)') \
                .select(from_json("data", ticker_schema).alias("data")) \
                .select('data.event_time','data.ticker','data.trade','data.volume',
 'data.price')
```

2. Daten in einem taumelnden Fenster verarbeiten. Im folgenden Beispiel werden die Daten auf der Grundlage des Eingabefelds „event_time“ in 10-minütigen taumelnden Fenstern gruppiert und die Ausgabe in einen Amazon-S3-Data-Lake geschrieben.

```
grouped_df = parsed_df \
                .groupBy(window("event_time", "10 minutes"), "city") \
                .agg(sum("clicks").alias("total_clicks"))

summary_df = grouped_df \
                .withColumn("window_start_time", col("window.start")) \
                .withColumn("window_end_time", col("window.end")) \
                .withColumn("year", year("window_start_time")) \
                .withColumn("month", month("window_start_time")) \
                .withColumn("day", dayofmonth("window_start_time")) \
                .withColumn("hour", hour("window_start_time")) \
                .withColumn("minute", minute("window_start_time")) \
                .drop("window")

write_result = summary_df \
                .writeStream \
                .format("parquet") \
                .trigger(processingTime="10 seconds") \
                .option("checkpointLocation", "s3a://bucket-stock-stream/stock-
stream-catalog-job/checkpoint/") \
                .option("path", "s3a://bucket-stock-stream/stock-stream-catalog-
job/summary_output/") \
                .partitionBy("year", "month", "day") \
                .start()
```

Gleitendes Fenster

Gleitende Fenster ähneln den taumelnden Fenstern, wobei die Fenster eine „feste Größe“ haben, aber sie können sich überlappen oder verschieben, solange die Laufzeit der Verschiebung kleiner ist als die Laufzeit des Fensters selbst. Es liegt in der Natur des Gleitens, dass eine Eingabe an mehrere Fenster gebunden werden kann.



Zum besseren Verständnis nehmen wir das Beispiel einer Bank, die einen möglichen Kreditkartenbetrug aufdecken möchte. Eine Streaming-Anwendung könnte einen kontinuierlichen Datenstrom von Kreditkartentransaktionen überwachen. Diese Transaktionen könnten in 10-minütigen Zeitfenstern zusammengefasst werden. Alle 5 Minuten würde das Fenster nach vorne gleiten, wobei die ältesten 5 Minuten an Daten gelöscht und die neuesten 5 Minuten an neuen Daten hinzugefügt würden. Innerhalb jedes Fensters können die Transaktionen nach Ländern gruppiert werden, um verdächtige Muster zu erkennen, z. B. eine Transaktion in den USA, die unmittelbar von einer anderen in Australien gefolgt wird. Zur Vereinfachung wollen wir nun solche Transaktionen als Betrug einstufen, wenn der Gesamtbetrag der Transaktionen über 100 USD liegt. Wenn ein solches Muster erkannt wird, deutet dies auf einen möglichen Betrug hin und die Kreditkarte könnte eingefroren werden.

Das Kreditkartenverarbeitungssystem sendet für jede Karten-ID zusammen mit dem Land eine Reihe von Transaktionsereignissen an Kinesis. Ein AWS Glue Job führt die Analyse aus und erzeugt die folgende aggregierte Ausgabe.

window_start_time	window_end_time	card_last_four	country	total_amount
2023-07-10 17:00:00	2023-07-10 17:10:00	6544	US	85
2023-07-10 17:00:00	2023-07-10 17:10:00	6544	Australien	10
2023-07-10 17:05:45	2023-07-10 17:15:45	6544	US	50
2023-07-10 17:10:45	2023-07-10 17:20:45	6544	US	50

window_start_time	window_end_time	card_last_four	country	total_amount
2023-07-10 17:10:45	2023-07-10 17:20:45	6544	Australien	150

Anhand der obigen Aggregation können Sie sehen, wie das 10-minütige Fenster alle 5 Minuten gleitet, summiert nach Transaktionsbetrag. Die Anomalie wird im Fenster von 17:10 bis 17:20 Uhr erkannt, wo es einen Ausreißer gibt, bei dem es sich um eine Transaktion für 150 USD in Australien handelt. AWS Glue kann diese Anomalie erkennen und mithilfe von boto3 ein Alarmereignis mit dem fraglichen Schlüssel zu einem SNS-Thema weiterleiten. Außerdem kann eine Lambda-Funktion dieses Thema abonnieren und Maßnahmen ergreifen.

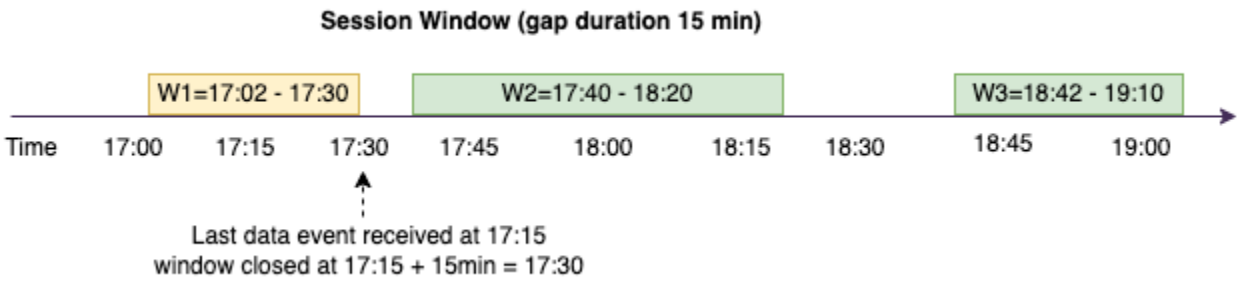
Daten in einem gleitenden Fenster verarbeiten

Die `group-by`-Klausel und die Fensterfunktion werden verwendet, um das gleitende Fenster wie unten gezeigt zu implementieren.

```
grouped_df = parsed_df \
            .groupBy(window(col("event_time"), "10 minute", "5 min"), "country",
 "card_last_four") \
            .agg(sum("tx_amount").alias("total_amount"))
```

Sitzungsfenster

Im Gegensatz zu den beiden oben genannten Fenstern, die eine feste Größe haben, kann das Sitzungsfenster eine statische oder dynamische Größe der Fensterlaufzeit haben, abhängig von den Eingaben. Ein Sitzungsfenster startet mit einem Eingabedaten-Ereignis und erweitert sich selbst, solange es innerhalb einer Lücke oder Inaktivitätsdauer Eingaben erhält.



Betrachten wir ein Beispiel. Das Unternehmen ABC Hotel möchte herausfinden, wann die stärkste Nachfrage in einer Woche herrscht, um seinen Gästen bessere Angebote machen zu können. Sobald ein Gast eincheckt, wird ein Sitzungsfenster gestartet und Spark behält dafür einen Aggregationsstatus bei. event-time-window Jedes Mal, wenn ein Gast eincheckt, wird ein Ereignis generiert und an dieses gesendet. Amazon Kinesis Data Streams Das Hotel entscheidet, dass das Hotel geschlossen werden event-time-window kann, wenn für einen Zeitraum von 15 Minuten kein Check-in stattfindet. Der nächste event-time-window wird wieder beginnen, wenn es einen neuen Check-in gibt. Die Ausgabe sieht wie folgt aus.

window_start_time	window_end_time	city	total_checkins
2023-07-10 17:02:00	2023-07-10 17:30:00	Dallas	50
2023-07-10 17:02:00	2023-07-10 17:30:00	Chicago	25
2023-07-10 17:40:00	2023-07-10 18:20:00	Dallas	75
2023-07-10 18:50:45	2023-07-10 19:15:45	Dallas	20

Der erste Check-in erfolgte um event_time=17.02. Die Aggregation beginnt um 17:02 Uhr. event-time-window Diese Aggregation wird so lange fortgesetzt, wie wir Ereignisse mit einer Dauer von 15 Minuten erhalten. Im obigen Beispiel war das letzte Ereignis, das wir erhalten haben, um 17.15 Uhr und in den nächsten 15 Minuten gab es keine Ereignisse. Infolgedessen hat Spark den Vorgang um 17:15 +15 Minuten = 17:30 Uhr geschlossen und event-time-window auf 17:02 bis 17:30 Uhr eingestellt. Es begann um 17:47 event-time-window Uhr neu, als es ein neues Check-In-Datenereignis empfing.

Daten in einem Sitzungsfenster verarbeiten

Die `group-by`-Klausel und die Fensterfunktion werden verwendet, um das gleitende Fenster zu implementieren.

```
grouped_df = parsed_df \
            .groupBy(session_window(col("event_time"), "10 minute"), "city") \
            .agg(count("check_in").alias("total_checkins"))
```

Ausgabemodi

Der Ausgabemodus ist der Modus, in dem die Ergebnisse der unbegrenzten Tabelle in die externe Sink geschrieben werden. Es sind drei Modi verfügbar. Im folgenden Beispiel zählt man das Auftreten eines Wortes, während die Datenzeilen in jedem Micro-Batch gestreamt und verarbeitet werden.

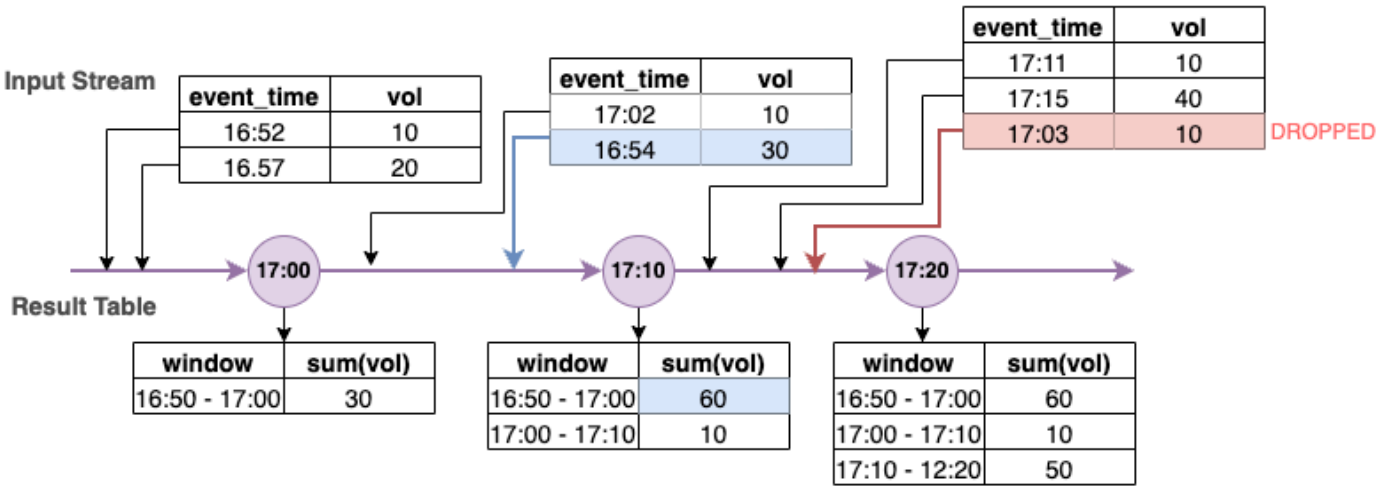
- Vollständiger Modus — Die gesamte Ergebnistabelle wird nach jeder Mikrostapelverarbeitung in die Senke geschrieben, auch wenn die Wortzahl in der aktuellen event-time-window Version nicht aktualisiert wurde.
- Anfügemodus – Dies ist der Standardmodus, bei dem nur die neuen Wörter und/oder Zeilen, die seit dem letzten Auslöser zur Ergebnistabelle hinzugefügt wurden, in die Sink geschrieben werden. Dieser Modus eignet sich für zustandsloses Streaming bei Abfragen wie map, flatMap, filter usw.
- Aktualisierungsmodus – Nur die Wörter und/oder Zeilen in der Ergebnistabelle, die seit dem letzten Auslöser aktualisiert oder hinzugefügt wurden, werden in die Sink geschrieben.

> **Note**
>
> Der Ausgabemodus = „Aktualisieren“ wird für Sitzungsfenster nicht unterstützt.

Umgang mit verspäteten Daten und Watermarks

Bei der Arbeit mit Echtzeitdaten kann es aufgrund von Netzwerklatenz und Upstream-Ausfällen zu Verzögerungen beim Eintreffen von Daten kommen. Wir benötigen einen Mechanismus, um die Aggregation bei den verpassten event-time-window Daten erneut durchzuführen. Dazu muss der Status jedoch aufrechterhalten werden. Gleichzeitig müssen die älteren Daten bereinigt werden, um die Größe des Zustands zu begrenzen. In Spark Version 2.1 wurde die Unterstützung für ein Feature namens Watermarking hinzugefügt, die den Status beibehält und es dem Benutzer ermöglicht, den Schwellenwert für verspätete Daten anzugeben.

Mit Bezug auf unser obiges Beispiel eines Börsentickers können wir davon ausgehen, dass der zulässige Schwellenwert für die verspäteten Daten nicht mehr als 10 Minuten beträgt. Der Einfachheit halber gehen wir von einem taumelnden Fenster aus, Ticker als AMZ, Handel als KAUFEN.



Im obigen Diagramm berechnen wir das Gesamtvolumen über ein 10-minütiges taumelndes Fenster. Der Auslöser erfolgt um 17.00, 17.10 und 17.20. Oberhalb des Zeitleistenpfeils befindet sich der Eingabedatenstrom und darunter die unbegrenzte Ergebnistabelle.

Im ersten 10-minütigen taumelnden Fenster haben wir auf der Grundlage von event_time aggregiert und das total_volume wurde als 30 berechnet. Im zweiten event-time-window Fall hat Spark das erste Datenereignis mit event_time= 17:02 empfangen. Da dies die maximale event_time ist, die Spark bisher gesehen hat, wird der Schwellenwert für das Watermark um 10 Minuten zurückgesetzt (d. h. watermark_event_time=16.52). Alle Datenereignisse mit einer event_time nach 16.52 werden für die zeitgebundene Aggregation berücksichtigt und alle Datenereignisse davor werden verworfen. Dadurch kann Spark einen Zwischenzustand für weitere 10 Minuten aufrechterhalten, um verspätete Daten zu verarbeiten. Um die Zeit 17.08 Uhr empfing Spark ein Ereignis mit einer event_time=16.54, das innerhalb des Schwellenwerts lag. Daher berechnete Spark den Wert „16:50 — 17:00“ neu event-time-window und das Gesamtvolumen wurde von 30 auf 60 aktualisiert.

Wenn Spark jedoch zur Auslösungszeit 17.20 ein Ereignis mit event_time=17.15 empfängt, setzt es die watermark_event_time=17.05. Daher wurde das verspätete Datenereignis mit event_time=17.03 als „zu spät“ betrachtet und ignoriert.

```
Watermark Boundary = Max(Event Time) - Watermark Threshold
```

Verwendung von Wasserzeichen in AWS Glue

Spark sendet oder schreibt die Daten erst dann in die externe Sink, wenn die Grenzwerte des Watermarks überschritten sind. Um ein Wasserzeichen zu implementieren in AWS Glue, siehe das Beispiel unten.

```
grouped_df = parsed_df \
                .withWatermark("event_time", "10 minutes") \
                .groupBy(window("event_time", "5 minutes"), "ticker") \
                .agg(sum("volume").alias("total_volume"))
```

Über die Überwachung von AWS Glue Streaming-Jobs

Die Überwachung Ihres Streaming-Auftrags ist ein entscheidender Teil des Aufbaus Ihrer ETL-Pipeline. Neben der Spark-Benutzeroberfläche können Sie auch Amazon verwenden, CloudWatch um die Metriken zu überwachen. Im Folgenden finden Sie eine Liste der vom AWS Glue Framework ausgegebenen Streaming-Metriken. Eine vollständige Liste aller AWS Glue Metriken finden Sie unter Überwachung AWS Glue mithilfe von CloudWatch Amazon-Metriken.

AWS Glue verwendet ein strukturiertes Streaming-Framework, um die Eingabeereignisse zu verarbeiten. Sie können entweder die Spark-API direkt in Ihrem Code verwenden oder den von `GlueContext` bereitgestellten `ForEachBatch` nutzen, der diese Metriken veröffentlicht. Um diese Metriken zu verstehen, müssen wir zuerst `windowSize` verstehen.

windowSize: `windowSize` ist das Micro-Batch-Intervall, das Sie angeben. Wenn Sie eine Fenstergröße von 60 Sekunden angeben, wartet der AWS Glue Streaming-Job 60 Sekunden (oder länger, wenn der vorherige Batch bis dahin noch nicht abgeschlossen ist), bevor er Daten in einem Batch aus der Streaming-Quelle liest und die unter angegebenen Transformationen anwendet. `ForEachBatch` Dies wird auch als Auslöser-Intervall bezeichnet.

Schauen wir uns die Metriken genauer an, um die Zustands- und Leistungsmerkmale zu verstehen.

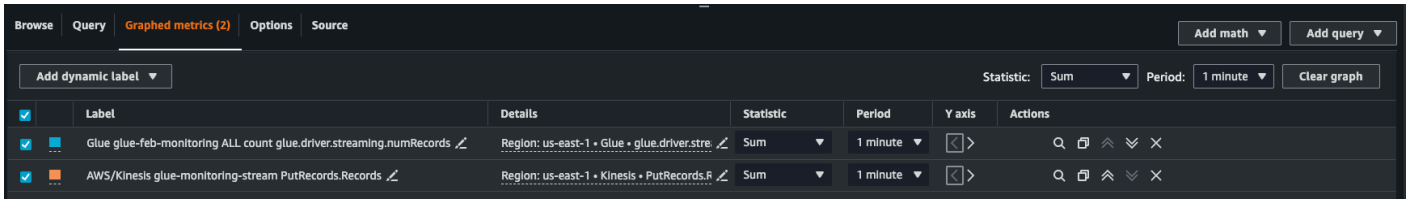
> **Note**
>
> Die Metriken werden alle 30 Sekunden ausgegeben. Wenn Ihre `windowSize` weniger als 30 Sekunden beträgt, handelt es sich bei den gemeldeten Metriken um eine Aggregation. Nehmen wir an, Ihre `windowSize` beträgt 10 Sekunden und Sie verarbeiten kontinuierlich 20 Datensätze pro Micro-Batch. In diesem Szenario wäre der ausgegebene Metrikwert für numRecords 60.
> Eine Metrik wird nicht ausgegeben, wenn dafür keine Daten verfügbar sind. Außerdem müssen Sie im Falle der Metrik für die Verzögerung von Verbrauchern das Feature aktivieren, um Metriken für sie zu erhalten.

AWS Glue Streaming-Metriken visualisieren

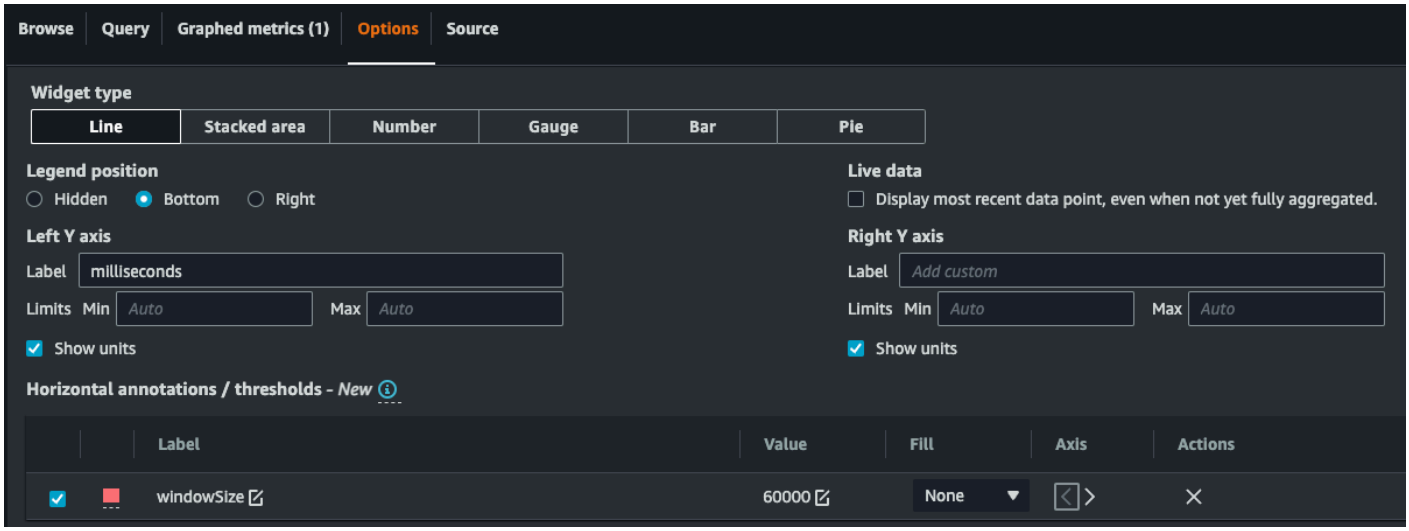
So zeichnen Sie visuelle Metriken auf:

1. Gehen Sie in der CloudWatch Amazon-Konsole zu Metrics und wählen Sie dann den Tab Durchsuchen. Wählen Sie dann unter „Benutzerdefinierte Namespaces“ die Option Glue.

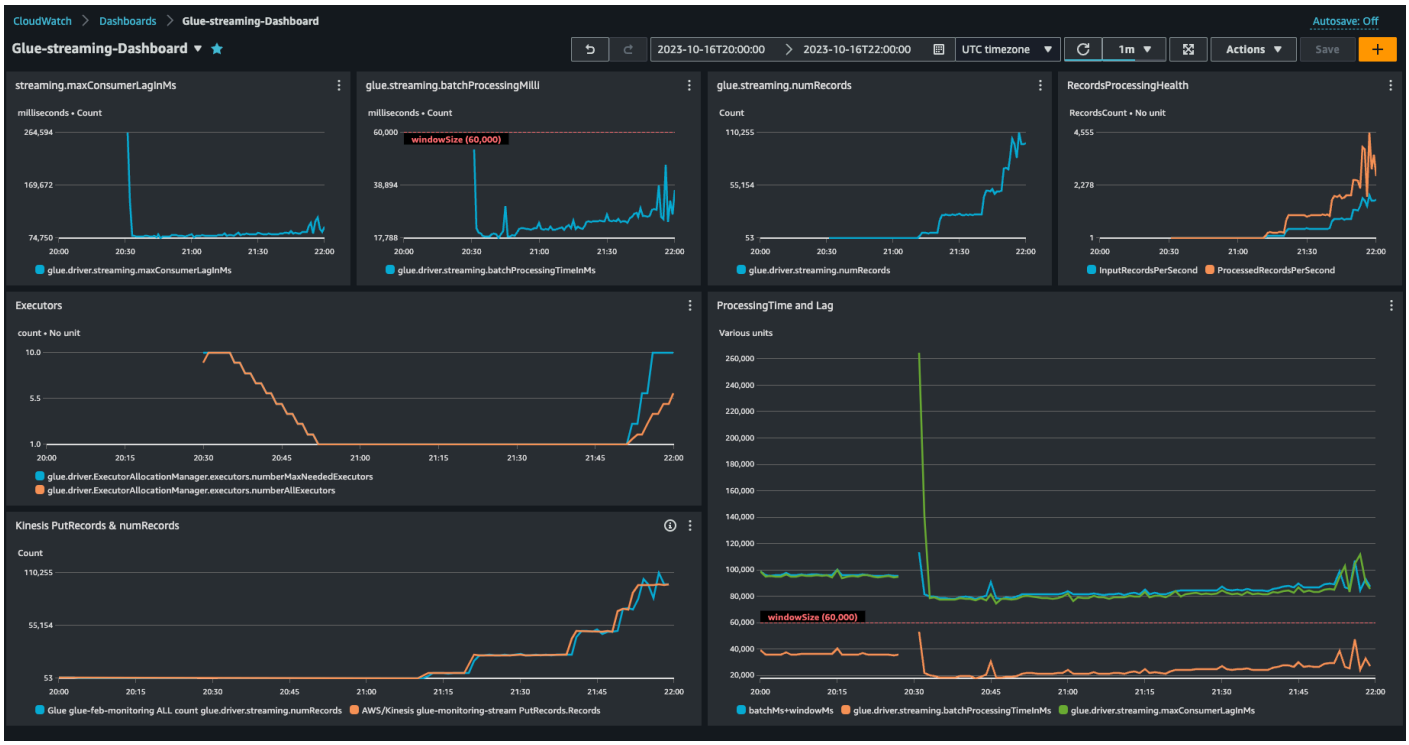
2. Wählen Sie Auftrag-Metriken, um Ihnen die Metriken für all Ihre Aufträge anzuzeigen.
3. Filtern Sie die Metriken nach Ihrem JobName = glue-feb-monitoring und dann nach JobRunId =ALL. Sie können auf das „+“-Zeichen klicken, wie in der Abbildung unten gezeigt, um es dem Suchfilter hinzuzufügen.
4. Wählen Sie das Kontrollkästchen für die Metriken aus, an denen Sie interessiert sind. In der folgenden Abbildung haben wir `numberAllExecutors` und `numberMaxNeededExecutors` ausgewählt.



5. Sobald Sie diese Metriken ausgewählt haben, können Sie zur Registerkarte Grafisch dargestellte Metriken wechseln und Ihre Statistiken anwenden.
6. Da die Metriken jede Minute ausgegeben werden, können Sie den „Durchschnitt“ über eine Minute für `batchProcessingTimeInMs` und `maxConsumerLagInMs` anwenden. Für `numRecords` kann die „Summe“ für jede Minute angewendet werden.
7. Sie können Ihrem Diagramm mithilfe der Registerkarte Optionen eine horizontale `windowSize`-Anmerkung hinzufügen.



8. Sobald Sie Ihre Metriken ausgewählt haben, erstellen Sie ein Dashboard und fügen Sie es hinzu. Hier ist ein Beispiel-Dashboard.

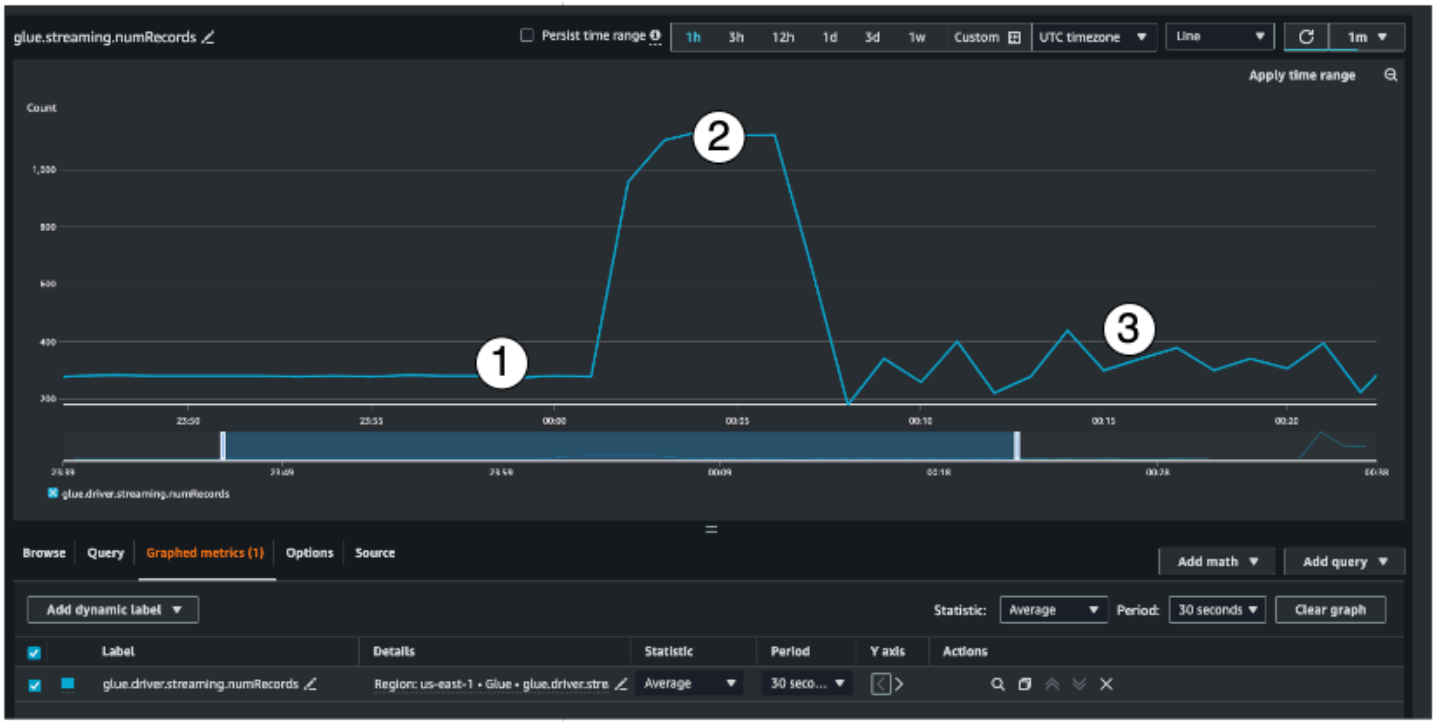


AWS Glue Streaming-Metriken verwenden

In diesem Abschnitt werden die einzelnen Metriken und ihre Beziehung zueinander beschrieben.

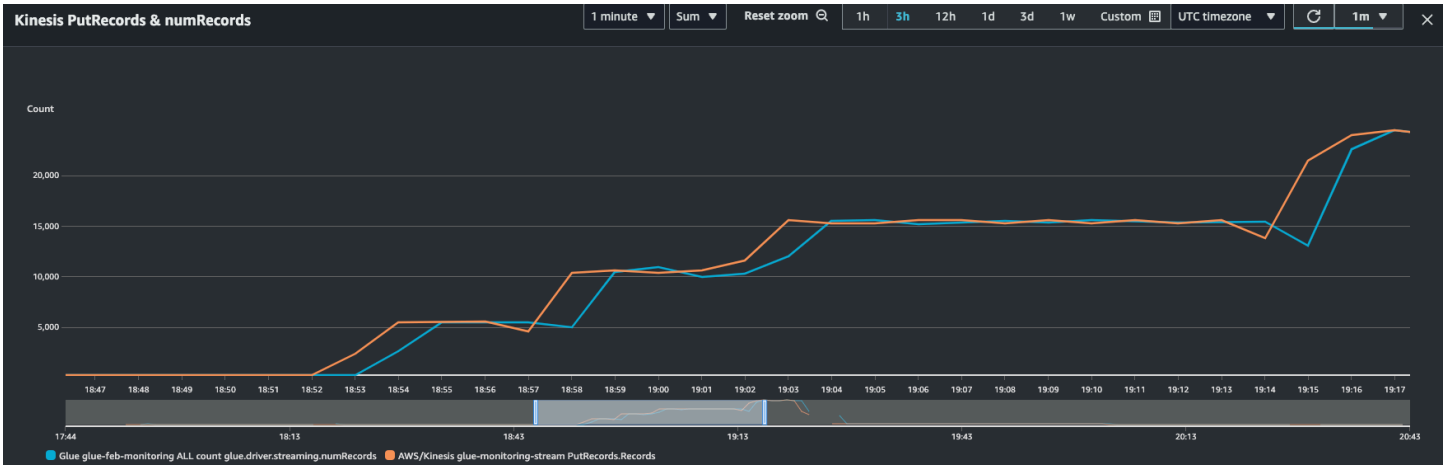
Anzahl der Datensätze (Metrik: streaming.numRecords)

Diese Metrik gibt an, wie viele Datensätze verarbeitet werden.



Diese Streaming-Metrik bietet gibt Aufschluss über die Anzahl der Datensätze, die Sie in einem Fenster verarbeiten. Zusammen mit der Anzahl der verarbeiteten Datensätze hilft es Ihnen auch, das Verhalten des Eingabe-Datenverkehrs zu verstehen.

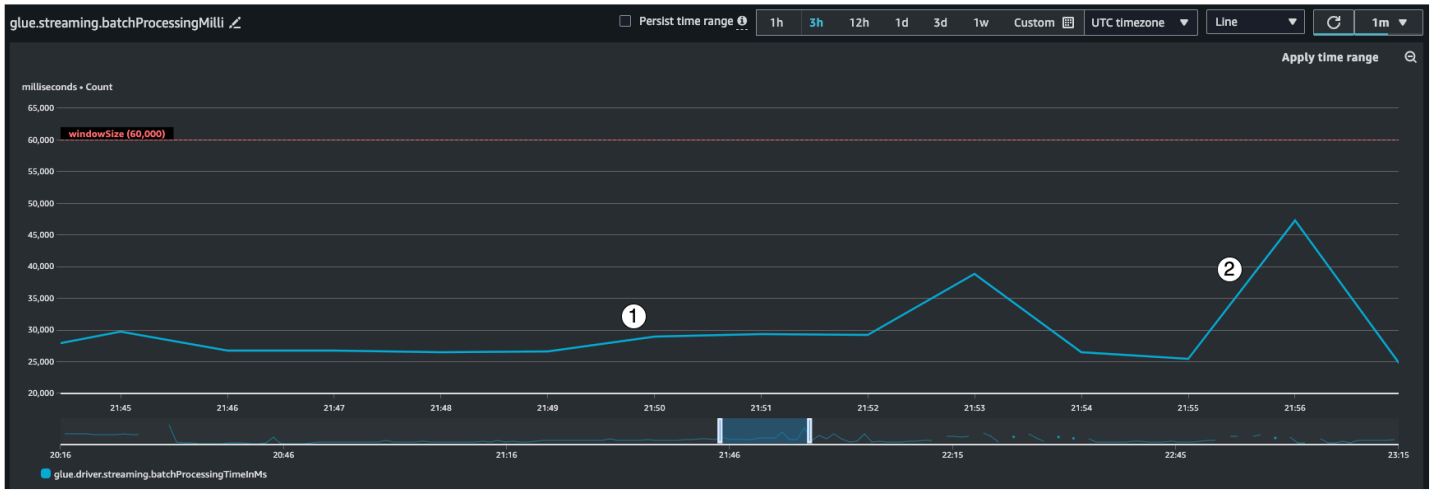
- Indikator 1 zeigt ein Beispiel für stabilen Datenverkehr ohne Spitzenbelastungen. In der Regel handelt es sich dabei um Anwendungen wie IoT-Sensoren, die in regelmäßigen Abständen Daten sammeln und diese an die Streaming-Quelle senden.
- Indikator 2 zeigt ein Beispiel für einen plötzlichen Anstieg des Datenverkehrs bei ansonsten stabiler Last. Dies kann in einer Clickstream-Anwendung passieren, wenn es ein Marketing-Ereignis wie den Black Friday gibt und die Anzahl der Klicks sprunghaft ansteigt.
- Indikator 3 zeigt ein Beispiel für unvorhersehbaren Datenverkehr. Unvorhersehbarer Verkehr bedeutet nicht, dass ein Problem vorliegt. Das liegt in der Natur der Eingabedaten. Um auf das Beispiel mit den IoT-Sensoren zurückzukommen: Sie können sich Hunderte von Sensoren vorstellen, die Ereignisse im Zusammenhang mit Wetteränderungen an die Streaming-Quelle senden. Da der Wetterwechsel nicht vorhersehbar ist, sind es auch die Daten nicht. Das Verständnis des Verkehrsaufkommens ist der Schlüssel zur Dimensionierung Ihrer Executors. Wenn die Eingangsdaten sehr sprunghaft sind, können Sie Auto Scaling in Betracht ziehen (mehr dazu später).



Sie können diese Metrik mit der PutRecords Kinesis-Metrik kombinieren, um sicherzustellen, dass die Anzahl der aufgenommenen Ereignisse und die Anzahl der gelesenen Datensätze nahezu identisch sind. Das ist vor allem dann nützlich, wenn Sie versuchen, Verzögerungen zu verstehen. Mit steigender Aufnahmerate steigt auch der Lesevorgang. `numRecords` AWS Glue

Batch-Verarbeitungszeit (Metrik: Streaming). batchProcessingTimeInMs)

Anhand der Metrik für die Batch-Verarbeitungszeit können Sie feststellen, ob der Cluster nicht ausreichend oder übermäßig ausgelastet ist.

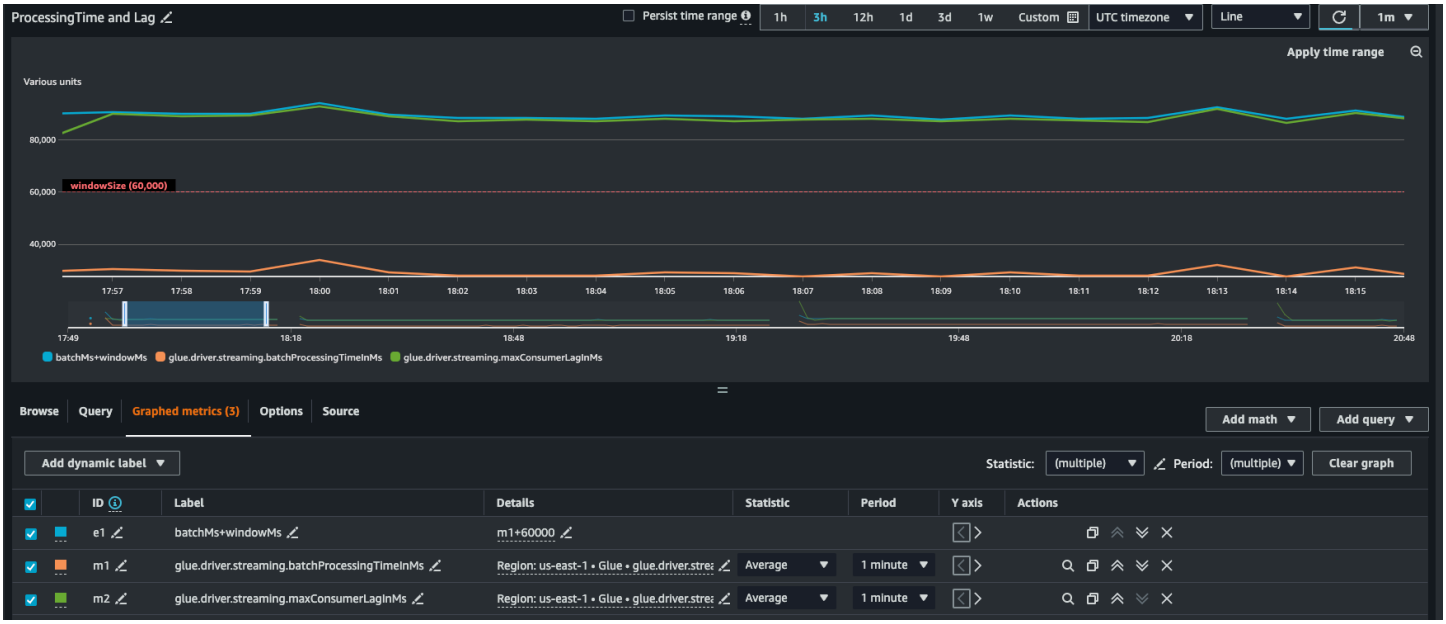


Diese Metrik gibt die Anzahl der Millisekunden an, die für die Verarbeitung eines jeden Micro-Batches von Datensätzen benötigt wurden. Das Hauptziel ist es, diese Zeit zu überwachen, um sicherzustellen, dass sie kürzer als das Intervall `windowSize` ist. Es ist in Ordnung, wenn die `batchProcessingTimeInMs` vorübergehend zu hoch ausfällt, solange sie sich im folgenden Fensterintervall wieder einpendelt. Indikator 1 zeigt eine mehr oder weniger stabile Zeit für die Bearbeitung des Auftrags an. Wenn jedoch die Anzahl der Eingabedatensätze

zunimmt, steigt auch die Zeit, die für die Verarbeitung des Auftrags benötigt wird, wie Indikator 2 zeigt. Wenn die `numRecords` nicht ansteigt, aber die Verarbeitungszeit ansteigt, müssen Sie die Auftragsverarbeitung der Executors genauer unter die Lupe nehmen. Es empfiehlt sich, einen Schwellenwert und einen Alarm einzustellen, um sicherzustellen, dass die `batchProcessingTimeInMs` nicht länger als 10 Minuten über 120 % bleibt. Weitere Informationen zum Einstellen von Alarmen finden Sie unter CloudWatch Amazon-Alarme verwenden.

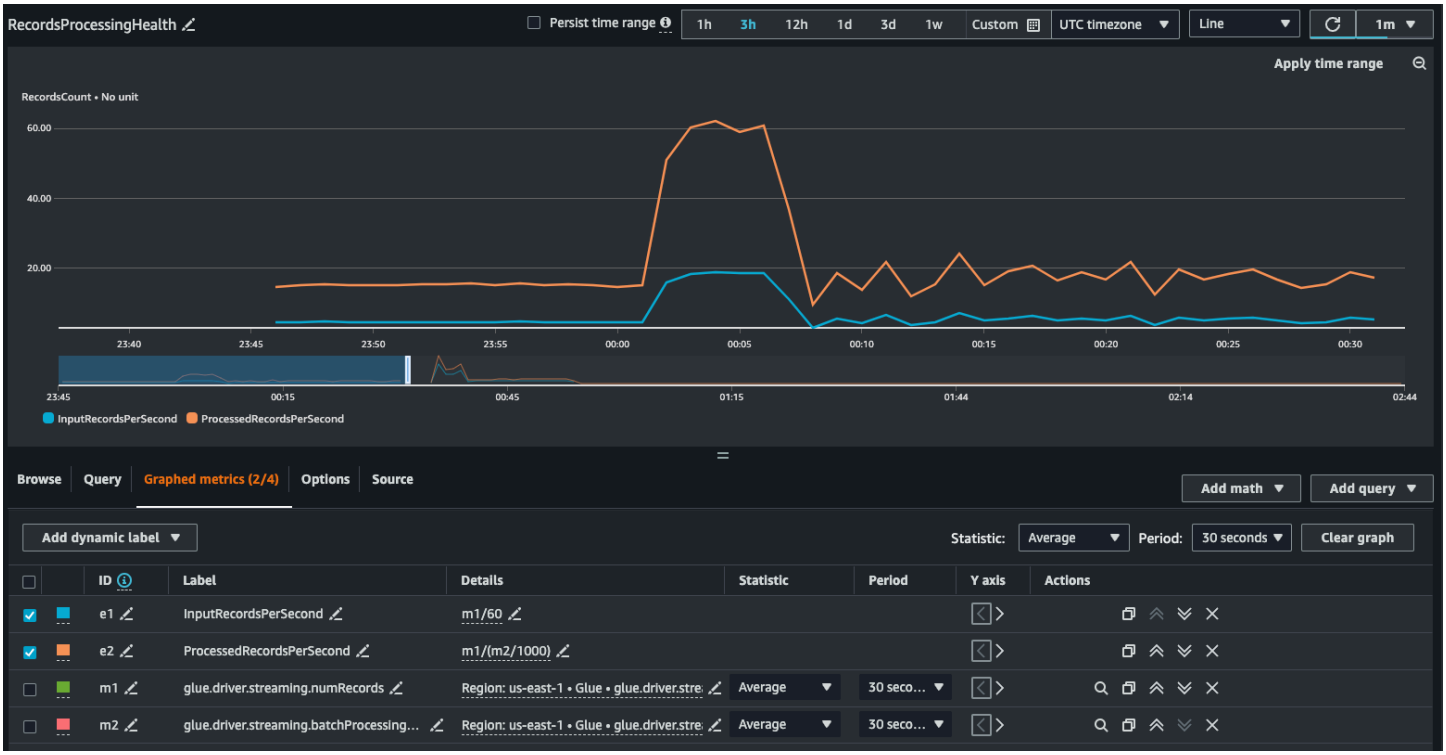
Verzögerung bei Verbrauchern (Metrik: Streaming). maxConsumerLagInMs)

Die Metrik für den Verbraucherverzögerungen hilft Ihnen zu verstehen, ob es eine Verzögerung bei der Verarbeitung von Ereignissen gibt. Wenn die Verzögerung zu groß ist, könnten Sie das Verarbeitungs-SLA, von dem Ihr Unternehmen abhängt, verpassen – auch wenn Sie eine korrekte windowSize haben. Sie müssen diese Metriken explizit über die `emitConsumerLagMetrics`-Verbindungsoption aktivieren. Weitere Informationen finden Sie unter KinesisStreamingSourceOptions.



Abgeleitete Metriken

Um tiefere Einblicke zu erhalten, können Sie abgeleitete Metriken erstellen, um mehr über Ihre Streaming-Jobs bei Amazon zu erfahren CloudWatch.



Sie können ein Diagramm mit abgeleiteten Metriken erstellen, um zu entscheiden, ob Sie mehr verwenden müssen DPUs. Auto Scaling hilft Ihnen dabei, dies automatisch zu tun, aber Sie können abgeleitete Metriken verwenden, um festzustellen, ob Auto Scaling effektiv funktioniert.

- InputRecordsPerSecondgibt die Geschwindigkeit an, mit der Sie Eingabedatensätze erhalten. Sie wird wie folgt abgeleitet: Anzahl der Eingabedatensätze (glue.driver.streaming.NumRecords)/. WindowSize
- ProcessingRecordsPerSecondgibt die Geschwindigkeit an, mit der Ihre Datensätze verarbeitet werden. Sie wird wie folgt abgeleitet: Anzahl der Eingabedatensätze (glue.driver.streaming.NumRecords)/. batchProcessingTime InMs

Wenn die Eingaberate höher ist als die Verarbeitungsrate, müssen Sie möglicherweise mehr Kapazität zur Verarbeitung Ihres Auftrags hinzufügen oder die Parallelität erhöhen.

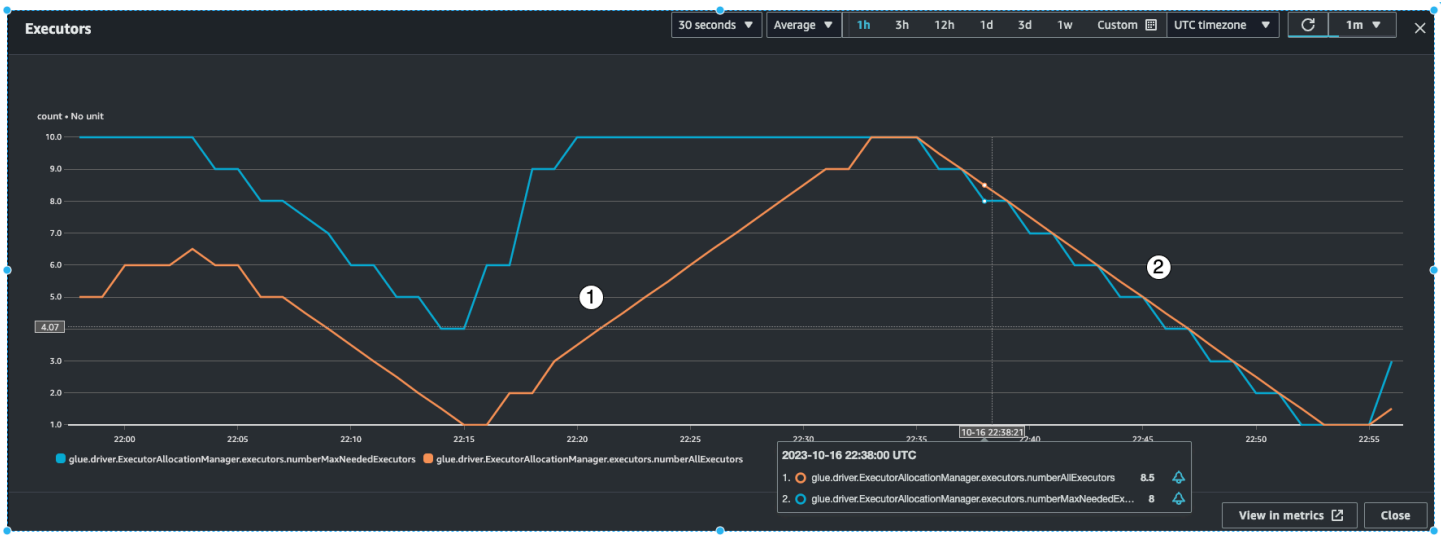
Auto-Scaling-Metriken

Wenn der Eingabe-Datenverkehr sprunghaft ansteigt, sollten Sie Auto Scaling aktivieren und die maximale Anzahl der Worker festlegen. Damit erhalten Sie zwei zusätzliche Metriken, `numberAllExecutors` und `numberMaxNeededExecutors`.

- numberAllExecutorsist die Anzahl der aktiv laufenden Job-Executoren

- numberMaxNeededExecutors ist die maximale Anzahl von (aktiv ausgeführten und ausstehenden) Job-Executoren, die benötigt werden, um die aktuelle Last zu bewältigen.

Diese beiden Metriken helfen Ihnen zu verstehen, ob Ihr Auto Scaling ordnungsgemäß funktioniert.



AWS Glue überwacht die `batchProcessingTimeInMs` Metrik über einige Mikrobatches und führt dabei eine von zwei Aktionen aus. Dabei werden die Executors aufskaliert, wenn `batchProcessingTimeInMs` näher an `windowSize` liegt, oder die Executors werden abskaliert, wenn `batchProcessingTimeInMs` vergleichsweise niedriger ist als `windowSize`. Außerdem wird ein Algorithmus zur schrittweisen Skalierung der Executors verwendet.

- Indikator 1 zeigt Ihnen, wie die aktiven Executors hochskaliert haben, um mit den maximal benötigten Executors mithalten zu können und die Last zu verarbeiten.
- Indikator 2 zeigt Ihnen, wie die aktiven Executors seit der niedrigen `batchProcessingTimeInMs` skaliert haben.

Sie können diese Metriken verwenden, um die aktuelle Parallelität auf Executor-Ebene zu überwachen und die Anzahl der maximalen Worker in Ihrer Auto-Scaling-Konfiguration entsprechend anzupassen.

So erzielen Sie die beste Leistung

Spark wird versuchen, eine zu lesende Aufgabe pro Shard im Amazon-Kinesis-Stream zu erstellen. Die Daten in jedem Shard werden zu einer Partition. Anschließend werden diese Aufgaben auf die Executors/Worker verteilt, abhängig von der Anzahl der Kerne auf jedem Worker (die Anzahl der

Kerne pro Worker hängt von dem von Ihnen gewählten Worker-Typ ab, `G.025X`, `G.1X`, usw.). Es ist jedoch nicht bestimmbar, wie die Aufgaben verteilt werden. Alle Aufgaben werden parallel auf ihren jeweiligen Kernen ausgeführt. Wenn es mehr Shards als verfügbare Executor-Kerne gibt, werden die Aufgaben in eine Warteschlange gestellt.

Sie können eine Kombination aus den oben genannten Metriken und der Anzahl der Shards verwenden, um Ihre Executors für eine stabile Last mit etwas Spielraum für Spitzenlasten bereitzustellen. Es empfiehlt sich, einige Iterationen Ihres Auftrags durchzuführen, um die ungefähre Anzahl der Worker zu ermitteln. Für einen instabilen/schwankenden Workload können Sie dasselbe tun, indem Sie Auto Scaling und maximale Worker einrichten.

Stellen Sie die `windowSize` gemäß den SLA-Anforderungen Ihres Unternehmens ein. Wenn Ihr Unternehmen beispielsweise verlangt, dass die verarbeiteten Daten nicht älter als 120 Sekunden sein dürfen, dann setzen Sie Ihre `windowSize` auf mindestens 60 Sekunden, so dass die durchschnittliche Verzögerung beim Verbraucher weniger als 120 Sekunden beträgt (siehe Abschnitt über die Verzögerung beim Verbraucher oben). Von dort aus können Sie, je nach Anzahl `numRecords` und Anzahl der Shards, die Kapazität so einplanen DPUs , dass Sie die `windowSize` meiste Zeit weniger als 70% Ihrer Shards haben. `batchProcessingTimeInMs`

> **Note**
>
> Hot Shards können zu Datenverzerrungen führen, was bedeutet, dass einige Shards/Partitionen viel größer sind als andere. Dies kann dazu führen, dass einige Aufgaben, die parallel ausgeführt werden, mehr Zeit benötigen und Nachzügleraufgaben verursachen. Dies hat zur Folge, dass der nächste Batch erst dann beginnen kann, wenn alle Aufgaben des vorherigen Batches abgeschlossen sind, was sich auf die `batchProcessingTimeInMillis` und die maximale Verzögerung auswirkt.

Null-ETL-Integrationen

Bei Zero-ETL handelt es sich um eine Reihe vollständig verwalteter Integrationen AWS , durch die der Aufbau von ETL-Daten-Pipelines für gängige Anwendungsfälle zur Erfassung und Replikation auf ein Minimum reduziert wird. Es stellt Daten aus verschiedenen Betriebs-, Transaktions- und Anwendungsquellen in Amazon SageMaker Lakehouse und Amazon Redshift zur Verfügung. Mit der Zero-ETL-Integration verfügen Sie über aktuellere Daten für Analysen, KI/ML und Berichte. Sie erhalten genauere und zeitnahe Einblicke für Anwendungsfälle wie Geschäfts-Dashboards, optimiertes Spielerlebnis, Überwachung der Datenqualität und Analyse des Kundenverhaltens. Sie können datengestützte Prognosen mit größerer Sicherheit treffen, das Kundenerlebnis verbessern und datengestützte Erkenntnisse im gesamten Unternehmen fördern.

Amazon Redshift ist ein schneller, vollständig verwalteter Data-Warehouse-Service für Datenmengen im Petabyte-Bereich, mit dem Sie im Zusammenspiel mit Ihren vorhandenen Business-Intelligence-Tools alle Ihre Daten einfach, effizient und wirtschaftlich analysieren können.

Amazon SageMaker Lakehouse vereinheitlicht all Ihre Daten in Amazon Simple Storage Service (S3) Data Lakes und Amazon Redshift Data Warehouses und hilft Ihnen, leistungsstarke Analysen und AI/ML Anwendungen auf einer einzigen Datenkopie zu erstellen. SageMaker Lakehouse bietet Ihnen die Flexibilität, mit allen Apache Iceberg-kompatiblen Tools und Engines direkt auf Ihre Daten zuzugreifen und sie abzufragen. Mit SageMaker Lakehouse haben Sie außerdem die Flexibilität, mit Apache Iceberg-kompatiblen Tools und Engines direkt auf Ihre Daten zuzugreifen und sie abzufragen. Darüber hinaus können Sie Ihre Daten mit integrierten, detaillierten Zugriffskontrollen schützen, die für alle Ihre Daten in allen Analysetools und -Engines gelten. Definieren Sie Berechtigungen einmal und geben Sie Daten vertrauensvoll innerhalb Ihres Unternehmens weiter.

Zero-ETL-Funktionen in AWS Glue

Zero-ETL-Integrationen AWS Glue vereinfachen die Datenaufnahme und Replikation von AWS Datendiensten und Drittanbieteranwendungen zu Zielen. AWS

AWS Zu den Diensten, die von Zero-ETL-Quellen unterstützt werden, gehören: AWS Glue

- Amazon-DynamoDB

Zu den von Zero-ETL unterstützten Drittanbieteranwendungen gehören:

- Facebook-Werbung
- Instagram-Werbung
- Salesforce
- Kundenbindung mit Salesforce Marketing Cloud
- SAP OData
- ServiceNow
- Zendesk
- Zoho CRM

AWS Zu den Dienstleistungen, die von Zero-ETL-Zielen unterstützt werden, gehören: AWS Glue

- Amazon Redshift
- SageMaker Amazonas-Seehaus

> **Note**
>
> Bei der Erstellung einer Zero-ETL-Integration mit einer Amazon DynamoDB DynamoDB-Quelle in AWS Glue wird das Ziel von Amazon Lakehouse unterstützt. SageMaker

Voraussetzungen für die Einrichtung einer Zero-ETL-Integration

Für die Einrichtung einer Integration zwischen Quelle und Ziel sind einige Voraussetzungen erforderlich, z. B. die Konfiguration von IAM-Rollen, mit denen AWS Glue auf Daten von der Quelle zugegriffen und in das Ziel geschrieben wird, sowie die Verwendung von KMS-Schlüsseln zum Verschlüsseln der Daten im Zwischen- oder Zielspeicherort.

Themen

- Quellressourcen einrichten
- Zielressourcen einrichten
- Ein Amazon Redshift Data Warehouse erstellen
- Einrichtung einer VPC für Ihre Zero-ETL-Integration
- Einrichtung einer kontenübergreifenden Zero-ETL-Integration

Quellressourcen einrichten

Führen Sie die folgenden Einrichtungsaufgaben nach Bedarf für Ihre Quelle aus.

Die Quellrolle einrichten

In diesem Abschnitt wird beschrieben, wie Sie eine Quellrolle übergeben, damit die Zero-ETL-Integration auf Ihre Verbindung zugreifen kann. Dies gilt auch nur für SaaS-Quellen.

> **Note**
>
> Um den Zugriff auf nur wenige Verbindungen zu beschränken, können Sie zunächst die Verbindung erstellen, um den Verbindungs-ARN zu erhalten. Siehe Konfiguration einer Quelle für eine Zero-ETL-Integration.

Erstellen Sie eine Rolle, die über die Berechtigungen für den Zugriff auf die Verbindung für die Integration verfügt:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GlueConnections",
            "Effect": "Allow",
            "Action": [
                "glue:GetConnections",
                "glue:GetConnection"
            ],
            "Resource": [
                "arn:aws:glue:*:<accountId>:catalog",
                "arn:aws:glue:us-east-1:<accountId>:connection/*"
            ]
        },
        {
            "Sid": "GlueActionBasedPermissions",
            "Effect": "Allow",
            "Action": [
                // Fetch entities:
                "glue:ListEntities",
                // Refresh connection credentials:
                "glue:RefreshOAuth2Tokens"
```

```
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Vertrauensrichtlinie:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "glue.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Zielressourcen einrichten

Führen Sie die folgenden Einrichtungsaufgaben je nach Bedarf für das AWS Glue Data Catalog- oder Amazon Redshift Data Warehouse-Integrationsziel aus.

Für Integrationen mit einem AWS Glue Datenbankziel:

- Eine Datenbank einrichten AWS Glue
- Bereitstellung einer RBAC-Richtlinie (Resource Based Access) für den Katalog
- Eine IAM-Zielrolle erstellen

Für Integrationen mit einem Amazon Redshift Datawarehouse-Ziel:

- https://docs.aws.amazon.com/glue/latest/dg/zero-etl-prerequisites.html#zero-etl-setup-target-redshift-data-warehouse

Eine Datenbank einrichten AWS Glue

Für Integrationen, die eine AWS Glue Datenbank verwenden:

So richten Sie eine Zieldatenbank im AWS Glue Datenkatalog mit einem Amazon S3 S3-Standort ein:

1. Wählen Sie auf der Startseite der AWS Glue Konsole unter Datenkatalog die Option Datenbank aus.
2. Wählen Sie in der oberen rechten Ecke Datenbank hinzufügen. Wenn Sie bereits eine Datenbank erstellt haben, stellen Sie sicher, dass der Speicherort mit Amazon S3 S3-URI für die Datenbank festgelegt ist.
3. Geben Sie einen Namen und einen Standort ein (Amazon S3 S3-URI). Beachten Sie, dass der Standort für die Zero-ETL-Integration erforderlich ist. Wenn Sie fertig sind, klicken Sie auf Datenbank erstellen.

> **Note**
>
> Der Amazon S3 S3-Bucket muss sich in derselben Region wie die AWS Glue Datenbank befinden.

Informationen zum Erstellen einer neuen Datenbank in AWS Glue finden Sie unter Erste Schritte mit dem AWS Glue Datenkatalog.

Sie können die Datenbank auch mit der `create-database`CLI erstellen AWS Glue. Beachten Sie, dass `LocationUri` die `--database-input` Eingabe erforderlich ist.

Optimieren von Iceberg-Tabellen

Sobald eine Tabelle AWS Glue in der Zieldatenbank erstellt wurde, können Sie die Komprimierung aktivieren, um Abfragen in Amazon Athena zu beschleunigen. Informationen zum Einrichten der Ressourcen (IAM-Rolle) für die Komprimierung finden Sie unter Voraussetzungen für die Tabellenoptimierung.

Weitere Informationen zum Einrichten der Komprimierung für die durch die Integration erstellte AWS Glue Tabelle finden Sie unter Optimieren von Iceberg-Tabellen.

Bereitstellung einer RBAC-Richtlinie (Resource Based Access) für den Katalog

Für Integrationen, die eine AWS Glue Datenbank verwenden, fügen Sie der Katalog-RBAC-Richtlinie die folgenden Berechtigungen hinzu, um Integrationen zwischen Quelle und Ziel zu ermöglichen.

> **Note**
>
> Für kontoübergreifende Integrationen müssen sowohl die Alice-Rollenrichtlinie (der Benutzer, der die Integration erstellt hat) als auch die Katalogressourcenrichtlinie die Ressource zulassen. `glue:CreateInboundIntegration` Für dasselbe Konto ist entweder eine Ressourcenrichtlinie oder eine Rollenrichtlinie ausreichend, die die Ressource zulässt`glue:CreateInboundIntegration`. Beide Szenarien müssen dies dennoch zulassen`glue.amazonaws.com`. `glue:AuthorizeInboundIntegration`

Sie können unter Datenkatalog auf die Katalogeinstellungen zugreifen. Geben Sie dann die folgenden Berechtigungen ein und geben Sie die fehlenden Informationen ein.

```
{
  "Version": "2012-10-17",
  "Statement": [
    { // Allow Alice to create Integration on Target Database
      "Principal": {
        "AWS": [
          "arn:aws:iam::<source-account-id>:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:CreateInboundIntegration"
      ],
      "Resource": [
        "arn:aws:glue:<region>:<Target-Account-Id>:catalog",
        "arn:aws:glue:<region>:<Target-Account-Id>:database/DatabaseName"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceArn": "arn:aws:dynamodb:<region>:<Account>:table/<table-name>"
        }
      }
    },
```

```
    { // Allow Glue to Authorize the Inbound Integration on behalf of Bob
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:AuthorizeInboundIntegration"
      ],
      "Resource": [
        "arn:aws:glue:<region>:<Target-Account-Id>:catalog",
        "arn:aws:glue:<region>:<Target-Account-Id>:database/DatabaseName"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "arn:aws:dynamodb:<region>:<account-id>:table/<table-name>"
        }
      }
    }
  ]
}
```

Eine IAM-Zielrolle erstellen

Erstellen Sie eine IAM-Zielrolle mit den folgenden Berechtigungen und Vertrauensbeziehungen:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::<target iceberg table s3 bucket>",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
```

```
            "Resource": "arn:aws:s3:::<target iceberg table s3 bucket>/prefix/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "glue:GetDatabase"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<account-id>:catalog",
                "arn:aws:glue:<region>:<account-id>:database/DatabaseName"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "glue:CreateTable",
                "glue:GetTable",
                "glue:UpdateTable",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:GetResourcePolicy"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<account-id>:catalog",
                "arn:aws:glue:<region>:<account-id>:database/<DatabaseName>",
                "arn:aws:glue:<region>:<account-id>:table/<DatabaseName>/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/Glue/ZeroETL"
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "logs:CreateLogGroup",
```

```
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Fügen Sie die folgende Vertrauensrichtlinie hinzu, damit der AWS Glue Dienst die Rolle übernehmen kann:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Ein Amazon Redshift Data Warehouse erstellen

Wenn Ihr Zero-ETL-Integrationsziel ein Amazon Redshift Redshift-Data Warehouse ist, erstellen Sie das Data Warehouse, falls Sie noch keines haben. Informationen zum Erstellen einer serverlosen Amazon Redshift Redshift-Arbeitsgruppe finden Sie unter Erstellen einer Arbeitsgruppe mit einem Namespace. Informationen zum Erstellen eines Amazon Redshift Redshift-Clusters finden Sie unter Cluster erstellen.

Für die Amazon Redshift Zielarbeitsgruppe oder den Ziel-Cluster muss der `enable_case_sensitive_identifier` Parameter aktiviert sein, damit die Integration erfolgreich ist. Weitere Informationen zur Aktivierung der Groß- und Kleinschreibung finden Sie unter

Aktivieren der Groß- und Kleinschreibung für Ihr Data Warehouse im Amazon Redshift Redshift-Managementleitfaden.

Nachdem die Einrichtung der Amazon Redshift Arbeitsgruppe oder des Clusters abgeschlossen ist, müssen Sie Ihr Data Warehouse konfigurieren. Weitere Informationen finden Sie im Amazon Redshift Management-Handbuch unter Erste Schritte mit Zero-ETL-Integrationen.

Einrichtung einer VPC für Ihre Zero-ETL-Integration

So richten Sie eine VPC für Ihre Zero-ETL-Integration ein:

1. Gehen Sie zu VPC > Your VPCs und wählen Sie Create VPC aus.
   a. Wählen Sie VPC und mehr.
   b. Legen Sie Ihren VPC-Namen fest.
   c. Stellen Sie den IPv4 CIDR ein: 10.0.0.0/16.
   d. Stellen Sie die Nummer von AZ auf 1 ein.
   e. Stellen Sie die Anzahl der öffentlichen und privaten Subnetze auf 1 ein.
   f. Stellen Sie NAT-Gateways auf Keine ein.
   g. Stellen Sie VPC-Endpunkte auf S3 Gateway ein.
   h. Aktivieren Sie DNS-Hostnamen und DNS-Auflösung.
2. Gehen Sie zu Endpoints und wählen Sie Create Endpoint aus.
3. Erstellen Sie Endpunkte für diese Dienste im privaten Subnetz Ihrer VPC (verwenden Sie die Standardsicherheitsgruppe):
   a. com.amazonaws.us-east-1.lambda
   b. com.amazonaws.us-east-1.glue
   c. com.amazonaws.us-east-1.sts

Stellen Sie die Verbindung her: AWS Glue

1. Gehen Sie zu AWS Glue> Datenverbindungen und wählen Sie Verbindung erstellen.
2. Wählen Sie „Netzwerk“ aus.
3. Wählen Sie die VPC, das Subnetz (privat) und die Standardsicherheitsgruppe aus, die Sie erstellt haben.

Einrichtung der Zielrolle für die VPC

Die Zielrolle muss über folgende Berechtigungen verfügen (zusätzlich zu den anderen für ETI Zero-Integrations erforderlichen Berechtigungen):

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CustomerVpc",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:DeleteTags",
                "ec2:DescribeRouteTables",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "glue:GetConnection"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Einrichtung der Eigenschaften der Ziel-Bein-Ressource

Wenn Sie die CLI verwenden, setzen Sie die Eigenschaften der Ziel-Leg-Ressource auf die AWS Glue Zieldatenbank, die Sie erstellt haben. Übergeben Sie den ARN der Zielrolle sowie den AWS Glue Verbindungsnamen.

```
aws glue create-integration-resource-property \
--resource-arn arn:aws:glue:us-east-1:<account-id>:database/exampletarget \
```

```
--target-processing-properties '{"RoleArn" : "arn:aws:iam::<account-id>:role/example-
role", "ConnectionName":"example-vpc-3"}' \
--endpoint-url https://example.amazonaws.com --region us-east-1
```

Mögliche Client-Fehler

Im Folgenden sind mögliche Client-Fehler für eine mit einer VPC konfigurierte Integration aufgeführt.

Fehlermeldung	Aktion erforderlich
Die angegebene Rolle ist nicht berechtigt, Glue: GetConnection on connection auszuführen. Fügen Sie diese Berechtigung zur Rollenric htlinie hinzu und warten Sie dann, bis die Integration wiederhergestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Die angegebene Rolle ist nicht autorisiert, ec2: DescribeSubnets auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie hinzu und warten Sie dann, bis die Integration wiederher gestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Die angegebene Rolle ist nicht autorisiert, ec2: DescribeSecurityGroups auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie hinzu und warten Sie dann, bis die Integration wiederhergestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Die angegebene Rolle ist nicht autorisiert, ec2: DescribeVpcEndpoints auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie hinzu und warten Sie dann, bis die Integration wiederhergestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Die angegebene Rolle ist nicht autorisiert, ec2: DescribeRouteTables auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie	Aktualisieren Sie die Rollenrichtlinie

Fehlermeldung	Aktion erforderlich
hinzu und warten Sie dann, bis die Integration wiederhergestellt ist.	
Die angegebene Rolle ist nicht autorisiert, ec2:CreateTags auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie hinzu und warten Sie dann, bis die Integration wiederher gestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Die angegebene Rolle ist nicht autorisiert, ec2:CreateNetworkInterface auszuführen. Fügen Sie diese Berechtigung zur Rollenrichtlinie hinzu und warten Sie dann, bis die Integration wiederhergestellt ist.	Aktualisieren Sie die Rollenrichtlinie
Vorausgesetzt, das Verbindungssubnetz enthält keinen gültigen S3-Endpunkt oder kein gültiges NAT-Gateway. Aktualisieren Sie das Subnetz und warten Sie dann, bis die Integrati on wiederhergestellt ist.	VPC-Subnetz-Endpunkte aktualisieren
Das Verbindungssubnetz wurde nicht gefunden. Aktualisieren Sie das Verbindun gssubnetz und warten Sie dann, bis die Integration wiederhergestellt ist.	Verbindung aktualisieren AWS Glue
Die Verbindungssicherheitsgruppe wurde nicht gefunden. Aktualisieren Sie die Verbindun gssicherheitsgruppe und warten Sie dann, bis die Integration wiederhergestellt ist.	AWS Glue Verbindung aktualisieren
Über die bereitgestellte VPC-Verbindung kann keine Verbindung zu S3 hergestellt werden. Aktualisieren Sie die Subnetzkonfigurati onen und warten Sie dann, bis die Integration wiederhergestellt ist.	VPC-Subnetz-Endpunkte aktualisieren

Fehlermeldung	Aktion erforderlich
Über die bereitgestellte VPC-Verbindung kann keine Verbindung zu Lambda hergestellt werden. Aktualisieren Sie die Subnetzkonfigurationen und warten Sie dann, bis die Integration wiederhergestellt ist.	VPC-Subnetz-Endpunkte aktualisieren

Einrichtung einer kontenübergreifenden Zero-ETL-Integration

So richten Sie eine kontenübergreifende Zero-ETL-Integration ein:

1. Konfigurieren Sie eine Zielressourcenrichtlinie wie unter beschrieben. Bereitstellung einer RBAC-Richtlinie (Resource Based Access) für den Katalog Stellen Sie sicher, dass die Rolle des Quellkontos für die Zielressource ausdrücklich zulässig ist.
2. Vergewissern Sie sich, dass die Rolle des Quellkontos (die Rolle, mit der die Integration erstellt wurde) Folgendes hat:

   JSON

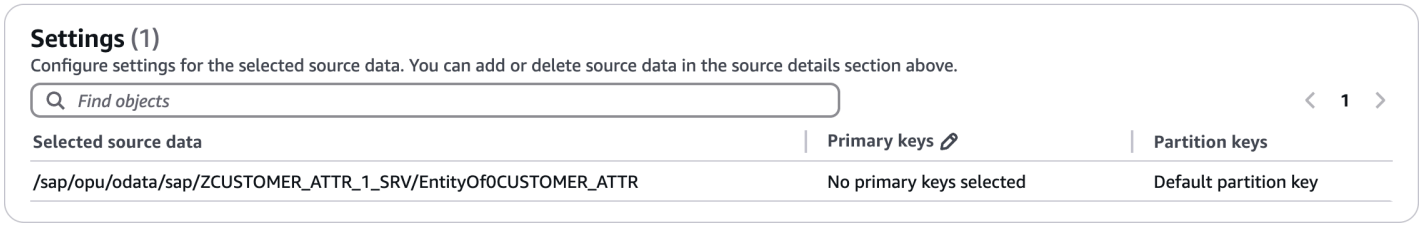
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt123456789012",
            "Action": [
                "glue:CreateInboundIntegration"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:glue:<region>:<target-account-id>:catalog",
                "arn:aws:glue:<region>:<target-account-id>:database/
DatabaseName"
            ]
        }
    ]
}
```

3. Erstellen Sie die Integration wie unter beschriebenEine Integration erstellen.

Konfiguration einer Quelle für eine Zero-ETL-Integration

Support für spezielle SAP-Entitäten

Für die meisten SaaS-Entitäten ermitteln wir die gültigen Primärschlüsselsätze während der Datenverarbeitung. Es gibt jedoch einige, für die ein zusätzlicher Schritt erforderlich ist, um den gültigen Primärschlüsselsatz als Eingabe bereitzustellen, insbesondere SAP-Entitäten, die mit beginnen`EntityOf`. Wenn eine `EntityOf` Entität ausgewählt wird, werden Sie angewiesen, den Satz von Primärschlüsseln anzugeben.



Konfiguration einer Amazon DynamoDB DynamoDB-Quelle

Um auf Daten aus Ihrer Amazon DynamoDB DynamoDB-Quelltabelle zuzugreifen, AWS Glue ist Zugriff erforderlich, um die Tabelle zu beschreiben und Daten aus ihr zu exportieren. Amazon DynamoDB hat kürzlich eine Funktion eingeführt, die die Konfiguration einer RBAC-Richtlinie (Resource Based Access) ermöglicht.

Das folgende Beispiel für eine Resource Based Access (RBAC) -Richtlinie verwendet einen Platzhalter (*) für die Integration:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "1111",
        "Effect": "Allow",
        "Principal": {
            "Service": "glue.amazonaws.com"
        },
        "Resource":"*",
        "Action": [
```

```
            "dynamodb:ExportTableToPointInTime",
            "dynamodb:DescribeTable",
            "dynamodb:DescribeExport"
        ],
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "<account-id>"
            },
            "StringLike": {
                "aws:SourceArn": "arn:aws:glue:<region>:<account-
id>:integration:*"
            }
        }
    }]
}
```

1. Fügen Sie für die DynamoDB, die Sie replizieren möchten, die obige RBAC-Richtlinienvorlage in die Tabelle Ressourcenbasierte Richtlinie für Tabelle ein und füllen Sie die Felder aus.
2. Wenn Sie die Richtlinie restriktiv gestalten möchten, müssen Sie die Richtlinie nach dem Erstellen der Integration aktualisieren und die vollständige `integrationArn` Richtlinie angeben und stattdessen die Bedingung verwenden. `StringEquals` `StringLike`
3. Stellen Sie sicher, dass Point-in-time Recovery (PITR) für die DynamoDB-Tabelle aktiviert ist.
4. Stellen Sie sicher, dass Sie `Describe Export` zur RBAC-Richtlinie (Resource Based Access) etwas hinzufügen.

Sie können die RBAC-Richtlinie auch mit dem folgenden Befehl zur Tabelle hinzufügen:

```
aws dynamodb put-resource-policy \
--resource-arn arn:aws:dynamodb:<region>:<account-id>:table/<ddb-table-name> \
--policy file://resource-policy-with-condition.json \
--region <region>
```

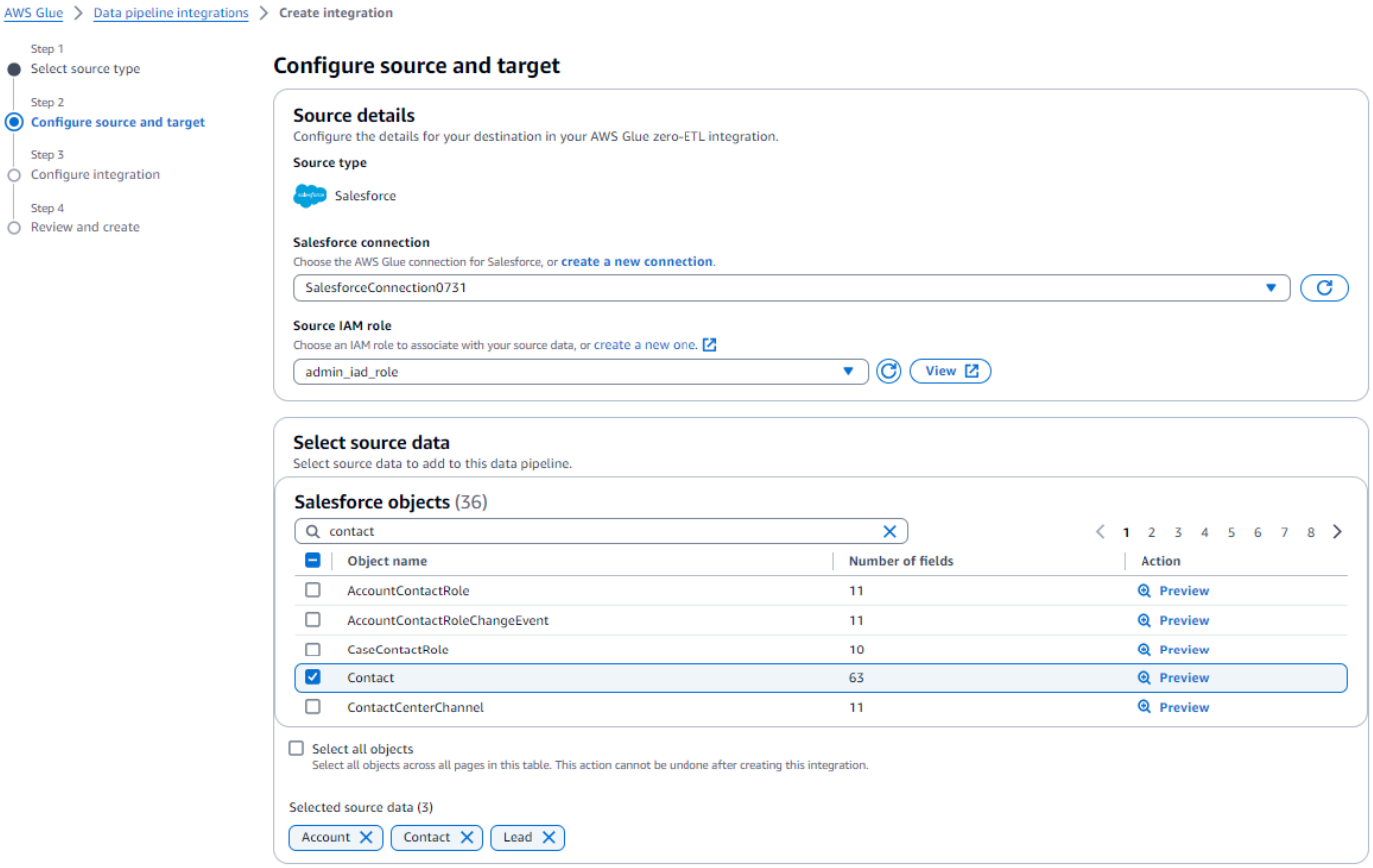
Um zu überprüfen, ob die Richtlinie korrekt angewendet wird, verwenden Sie den folgenden Befehl, um die Ressourcenrichtlinie für eine Tabelle abzurufen:

```
aws dynamodb get-resource-policy \
--resource-arn arn:aws:dynamodb:<region>:<account-id>:table/<ddb-table-name>
--region <region>
```

Konfiguration einer Salesforce-Quelle

Informationen zum Herstellen einer Verbindung für eine Salesforce-Quelle finden Sie unter Verbindung zu Salesforce herstellen.

Sobald Sie die Verbindung erstellt haben, können Sie die Quelldaten angeben, die repliziert werden sollen.



Mit Ihrer Zero-ETL-Integration können Sie DDL-Operationen für unterstützte Entitäten ausführen. Eine Liste der Entitäten, die nicht unterstützt werden, finden Sie unter. Nicht unterstützte Entitäten und Felder für Salesforce

Konfiguration einer Salesforce Marketing Cloud-Account-Engagement-Quelle

Informationen zum Herstellen einer Verbindung für eine Salesforce Marketing Cloud-Account-Engagement-Quelle finden Sie unter Verbindung mit Salesforce Marketing Cloud Account Engagement herstellen .

Mit Ihrer Zero-ETL-Integration können Sie DDL-Operationen für die folgenden unterstützten Entitäten ausführen:

Bezeichnung der Entität	Entity-Name
Kampagne	Kampagne
Liste	auflisten
Dynamischer Inhalt	dynamischer Inhalt
Mitgliedschaft auflisten	Mitgliedschaft auf der Liste
Interessent	Aussicht
Benutzer	user
EmailTemplate	E-Mail-Vorlage
EngagementStudioProgram	engagement-studio-program
Landingpage	Landingpage
E-Mail auflisten	E-Mail auflisten

Zusätzliche Salesforce-Konfiguration

Salesforce Zero-ETL benötigt die Lake Formation Formation-Berechtigung für die Glue-Datenbank, andernfalls wird es mit dem folgenden Fehler `IngestionFailed` aus dem Protokoll abgerufen:

```
"errorMessage": "Insufficient lake formation permissions on Target Glue database."
```

Konfiguration einer SAP-Quelle OData

Informationen zum Herstellen einer Verbindung für eine OData SAP-Quelle finden Sie unter Verbindung zu SAP herstellen OData .

Der OData SAP-Konnektor in einer Zero-ETL-Integration unterstützt keine Entitäten, die mit beginnen. `Entity0f`

Konfiguration einer Quelle ServiceNow

Informationen zum Herstellen einer Verbindung für eine ServiceNow Quelle finden Sie unter Verbindung herstellen zu ServiceNow .

Konfiguration einer Zendesk-Quelle

Informationen zum Herstellen einer Verbindung für eine Zendesk-Quelle finden Sie unter. Verbindung zu Zendesk herstellen

Mit Ihrer Zero-ETL-Integration können Sie die folgenden DDL-Operationen für unterstützte Entitäten ausführen:

Bezeichnung der Entität	Entity-Name	Unterstützt erstellen	Update wird unterstützt	Löschen wird unterstützt
Tickets	Eintrittskarten	Y	Y	Y
Benutzer	Benutzer	Y	Y	Y
Zufrieden heitsbewertung	Zufrieden heitsbewertung	Y	Y	N
Artikel	Artikel	Y	Y	N
Organisation	Organisationen	Y	Y	Y
Calls	Anrufe	Y	Y	N
Ruf Legs an	Beine	Y	Y	N

Konfiguration einer Zoho CRM-Quelle

Informationen zum Herstellen einer Verbindung für eine Zoho CRM-Quelle finden Sie unter. Verbindung zu Zoho CRM herstellen

Mit Ihrer Zero-ETL-Integration können Sie die folgenden DDL-Operationen für unterstützte Entitäten ausführen:

Bezeichnung der Entität	Entity-Name	DML-Insert wird unterstützt	DML-Modify wird unterstützt	DML-Delete wird unterstützt	DDL-Insert wird unterstützt	DDL-Modify wird unterstützt	DDL-Delete wird unterstützt
Führt	lead	Y	Y	Y	Y	Y	Y
Konten	Konto	Y	Y	Y	Y	Y	Y
Kontakte	contact	Y	Y	Y	Y	Y	Y
Kampagnen	Kampagne	Y	Y	Y	Y	Y	Y
Aufgaben	Aufgabe	Y	Y	Y	Y	Y	Y
--Ereignisse	event	Y	Y	Y	Y	Y	Y
Calls	Aufruf	Y	Y	Y	Y	Y	Y
Lösungen	Lösung	Y	Y	Y	Y	Y	Y
Produkte	Produkt	Y	Y	Y	Y	Y	Y
Anbieter	Lieferant	Y	Y	Y	Y	Y	Y
Zitate	Zitat	Y	Y	Y	Y	Y	Y
Verkaufsaufträge	Verkaufsauftrag	Y	Y	Y	Y	Y	Y
Bestellungen	Bestellung	Y	Y	Y	Y	Y	Y
Rechnungen	Rechnung	Y	Y	Y	Y	Y	Y
Fälle	Fall	Y	Y	Y	Y	Y	Y

Bezeichnung der Entität	Entity-Name	DML-Insert wird unterstützt	DML-Modify wird unterstützt	DML-Delete wird unterstützt	DDL-Insert wird unterstützt	DDL-Modify wird unterstützt	DDL-Delete wird unterstützt
Preisbücher	Preisbuch	Y	Y	Y	Y	Y	Y

Konfiguration einer Facebook-Werbequelle

Informationen zum Herstellen einer Verbindung für eine Facebook-Werbequelle finden Sie unter Verbindung zu Facebook-Werbeanzeigen herstellen .

Mit Ihrer Zero-ETL-Integration können Sie die folgenden DDL-Operationen für unterstützte Entitäten ausführen:

Bezeichnung der Entität	Entity-Name	Unterstützt erstellen	Update wird unterstützt	Löschen wird unterstützt
Adset	*/Vermögenswerte	Y	Y	Y
Kampagne	*/Kampagnen	Y	Y	Y
Werbeanzeigen	*/Anzeigen	Y	Y	Y

Konfiguration einer Instagram-Werbequelle

Informationen zum Herstellen einer Verbindung für eine Instagram-Werbequelle finden Sie unter Verbindung zu Instagram-Anzeigen herstellen .

Mit Ihrer Zero-ETL-Integration können Sie die folgenden DDL-Operationen für unterstützte Entitäten ausführen:

Entity-Name	Unterstützte Erstellung	Update wird unterstützt	Löschen wird unterstützt
*/Adsets	Y	Y	Y
*/Kampagnen	Y	Y	Y
*/Anzeigen	Y	Y	Y

Nicht unterstützte Entitäten und Felder für Salesforce

Die folgenden Salesforce-Entitäten oder -Felder werden für die Verwendung in einer Zero-ETL-Integration mit einer Salesforce-Quelle nicht unterstützt.

```
AccountChangeEvent, AccountContactRoleChangeEvent, AccountHistory, AccountShare,
 ActiveFeatureLicenseMetric, ActivePermSetLicenseMetric, ActiveProfileMetric,
 ActivityFieldHistory, amzsec__asi_Telemetry_Data_Store__ChangeEvent,
 amzsec__asi_Telemetry_Data_Store__History, amzsec__asi_Telemetry_Data_Store__Share,
 amzsec__asi_Telemetry_Job_Log__ChangeEvent, amzsec__asi_Telemetry_Job_Log__History,
 amzsec__asi_Telemetry_Job_Log__Share, amzsec__asi_Telemetry_Requirement__ChangeEvent,
 amzsec__asi_Telemetry_Requirement__History, amzsec__asi_Telemetry_Requirement__Share,
 ApexClass, ApexComponent, ApexLog, ApexPage, ApexTestQueueItem, ApexTestResult,
 ApexTrigger, AssetChangeEvent, AssetHistory, AssetRelationshipHistory,
 AssetShare, AssignmentRule, AssociatedLocationHistory, AsyncApexJob,
 AuditTrailFileExportShare, AuthorizationFormConsentChangeEvent,
 AuthorizationFormConsentHistory, AuthorizationFormConsentShare,
 AuthorizationFormDataUseHistory, AuthorizationFormDataUseShare,
 AuthorizationFormHistory, AuthorizationFormShare, AuthorizationFormTextHistory,
 AuthProvider, AuthSession, BatchJobHistory, BatchJobPartFailedRecordHistory,
 BatchJobPartHistory, BatchJobShare, BrandTemplate, BriefcaseAssignmentChangeEvent,
 BriefcaseDefinitionChangeEvent, BusinessBrandShare, BusinessHours,
 BusinessProcess, CalcMatrixColumnRangeHistory, CalcProcStepRelationshipHistory,
 CalculationMatrixColumnHistory, CalculationMatrixHistory, CalculationMatrixRowHistory,
 CalculationMatrixShare, CalculationMatrixVersionHistory, CalculationProcedureHistory,
 CalculationProcedureShare, CalculationProcedureStepHistory,
 CalculationProcedureVariableHistory, CalculationProcedureVersionHistory, Calendar,
 CalendarViewShare, CallCenter, CallCoachConfigModifyEvent, CampaignChangeEvent,
 CampaignHistory, CampaignMemberChangeEvent, CampaignMemberStatusChangeEvent,
 CampaignShare, CaseChangeEvent, CaseHistory, CaseHistory2 CaseHistory2ChangeEvent,
 CaseRelatedIssueChangeEvent, CaseRelatedIssueHistory, CaseShare, CaseStatus,
 CaseTeamMember, CaseTeamRole, CaseTeamTemplate, CaseTeamTemplateMember,
```

```
CaseTeamTemplateRecord, CategoryNode, ChangeRequestChangeEvent, ChangeRequestHistory,
ChangeRequestRelatedIssueChangeEvent, ChangeRequestRelatedIssueHistory,
ChangeRequestRelatedItemChangeEvent, ChangeRequestRelatedItemHistory,
ChangeRequestShare, ChatRetirementRdyMetrics, ChatterActivity, ClientBrowser,
CollaborationGroup, CollaborationGroupMember, CollaborationGroupMemberRequest,
CollaborationInvitation, CommSubscriptionChannelTypeHistory,
CommSubscriptionChannelTypeShare, CommSubscriptionConsentChangeEvent,
CommSubscriptionConsentHistory, CommSubscriptionConsentShare, CommSubscriptionHistory,
CommSubscriptionShare, CommSubscriptionTimingHistory, Community, ConnectedApplication,
ContactChangeEvent, ContactHistory, ContactPointAddressChangeEvent,
ContactPointAddressHistory, ContactPointAddressShare, ContactPointConsentChangeEvent,
ContactPointConsentHistory, ContactPointConsentShare, ContactPointEmailChangeEvent,
ContactPointEmailHistory, ContactPointEmailShare, ContactPointPhoneChangeEvent,
ContactPointPhoneHistory, ContactPointPhoneShare, ContactPointTypeConsentChangeEvent,
ContactPointTypeConsentHistory, ContactPointTypeConsentShare, ContactRequestShare,
ContactShare, ContentDocumentChangeEvent, ContentDocumentHistory,
ContentDocumentLink, ContentDocumentLinkChangeEvent, ContentDocumentSubscription,
ContentFolderItem, ContentFolderLink, ContentFolderMember, ContentNote,
ContentNotification, ContentTagSubscription, ContentUserSubscription,
ContentVersionChangeEvent, ContentVersionComment, ContentVersionHistory,
ContentVersionRating, ContentWorkspace, ContentWorkspaceMember,
ContentWorkspacePermission, ContentWorkspaceSubscription, ContractChangeEvent,
ContractHistory, ContractLineItemChangeEvent, ContractLineItemHistory,
ContractStatus, Conversation, ConversationParticipant, CronJobDetail,
CronTrigger, CustomBrand, CustomBrandAsset, CustomerShare, CustomHttpHeader,
DashboardComponent, DataUseLegalBasisHistory, DataUseLegalBasisShare,
DataUsePurposeHistory, DataUsePurposeShare, DecisionTableRecordset, DeleteEvent,
DocumentAttachmentMap, Domain, DomainSite, DTRecordsetReplicaShare, EmailBounceEvent,
EmailMessageChangeEvent, EmailServicesAddress, EmailServicesFunction,
EmailTemplate, EmailTemplateChangeEvent, EngagementAttendeeChangeEvent,
EngagementAttendeeHistory, EngagementChannelTypeHistory, EngagementChannelTypeShare,
EngagementInteractionChangeEvent, EngagementInteractionHistory,
EngagementInteractionShare, EngagementInterface, EngagementTopicChangeEvent,
EngagementTopicHistory, EntitlementChangeEvent, EntitlementHistory,
EntitlementTemplate, EntityMilestoneHistory, EntitySubscription, EventChangeEvent,
EventRelationChangeEvent, EventRelayConfigChangeEvent, ExpressionSetHistory,
ExpressionSetShare, ExpressionSetVersionHistory, ExternalEventMappingShare,
FeedAttachment, FeedLike, FeedPollChoice, FeedPollVote, FeedSignal,
FieldPermissions, FieldSecurityClassification, FiscalYearSettings,
FlowInterviewLogShare, FlowInterviewShare, FlowOrchestrationEvent,
FlowOrchestrationInstanceShare, FlowOrchestrationStageInstanceShare,
FlowOrchestrationStepInstanceShare, FlowOrchestrationWorkItemShare,
FlowRecordShare, FlowRecordVersionChangeEvent, FlowTestResultShare, Folder,
Group, GroupMember, Holiday, IdeaComment, IdpEventLog, ImageHistory, ImageShare,
```

```
IncidentChangeEvent, IncidentHistory, IncidentRelatedItemChangeEvent,
IncidentRelatedItemHistory, IncidentShare, IndividualChangeEvent,
IndividualHistory, IndividualShare, KnowledgeableUser, LeadChangeEvent, LeadHistory,
LeadShare, LeadStatus, LightningExitByPageMetrics, LightningToggleMetrics,
LightningUsageByAppTypeMetrics, LightningUsageByBrowserMetrics,
LightningUsageByFlexiPageMetrics, LightningUsageByPageMetrics, ListEmailChangeEvent,
ListEmailShare, ListView, LocationChangeEvent, LocationHistory, LocationShare,
LocationTrustMeasureShare, LoginHistory, LoginIp, MacroChangeEvent,
MacroHistory, MacroInstructionChangeEvent, MacroShare, MacroUsageShare,
ManagedContentVariantChangeEvent, MessagingEndUserHistory, MessagingEndUserShare,
MessagingSessionHistory, MessagingSessionShare, MilestoneType, MLEngagementEvent,
ObjectPermissions, OpportunityChangeEvent, OpportunityContactRoleChangeEvent,
OpportunityFieldHistory, OpportunityLineItemChangeEvent, OpportunityShare,
OpportunityStage, OrderChangeEvent, OrderHistory, OrderItemChangeEvent,
OrderItemHistory, OrderShare, OrderStatus, Organization, OrgEmailAddressSecurity,
OrgWideEmailAddress, OutgoingEmail, OutgoingEmailRelation, PackageLicense,
PartnerRole, PartyConsentChangeEvent, PartyConsentHistory, PartyConsentShare,
Period, PermissionSet, PermissionSetAssignment, PermissionSetTabSetting,
Pricebook2ChangeEvent, Pricebook2History, PricebookEntryChangeEvent,
PricebookEntryHistory, PrivacyJobSessionShare, PrivacyObjectSessionShare,
PrivacyRTBFRequestHistory, PrivacyRTBFRequestShare, PrivacySessionRecordFailureShare,
ProblemChangeEvent, ProblemHistory, ProblemIncidentChangeEvent,
ProblemIncidentHistory, ProblemRelatedItemChangeEvent, ProblemRelatedItemHistory,
ProblemShare, ProcessDefinition, ProcessExceptionEvent, ProcessExceptionShare,
ProcessInstanceChangeEvent, ProcessInstanceStep, ProcessInstanceStepChangeEvent,
ProcessNode, Product2ChangeEvent, Product2History, ProductEntitlementTemplate,
Profile, ProfileSkillEndorsementHistory, ProfileSkillHistory,
ProfileSkillShare, ProfileSkillUserHistory, PromptActionShare, PromptErrorShare,
QueueSobject, QuickTextChangeEvent, QuickTextHistory, QuickTextShare,
QuickTextUsageShare, RecentlyViewed, RecommendationChangeEvent,
RecordActionHistory, RecordAlertHistory, RecordAlertShare, RecordType,
ScorecardShare, SellerHistory, SellerShare, ServiceContractChangeEvent,
ServiceContractHistory, ServiceContractShare, SetupAuditTrail, SetupEntityAccess,
SharingRecordCollectionShare, Site, SiteHistory, SiteRedirectMapping,
SocialPersonaHistory, SocialPostChangeEvent, SocialPostHistory, SocialPostShare,
SolutionHistory, SolutionStatus, StaticResource, StreamingChannelShare,
TableauHostMappingShare, TaskChangeEvent, TaskPriority, TaskStatus,
ThreatDetectionFeedback, TimelineObjectDefinitionChangeEvent, TodayGoalShare,
Topic, TopicUserEvent, Translation, User, UserAppMenuCustomizationShare,
UserChangeEvent, UserDefinedLabelAssignmentShare, UserDefinedLabelShare,
UserEmailPreferredPersonShare, UserLicense, UserLogin, UserPackageLicense,
UserPreference, UserPrioritizedRecordShare, UserProvisioningRequestShare,
UserRole, UserShare, VideoCallChangeEvent, VideoCallParticipantChangeEvent,
VideoCallRecordingChangeEvent, VideoCallShare, VisualforceAccessMetrics,
```

```
 VoiceCallChangeEvent, VoiceCallRecordingChangeEvent, VoiceCallShare, Vote,
 WebLink, WorkAccessShare, WorkBadgeDefinitionHistory, WorkBadgeDefinitionShare,
 WorkOrderChangeEvent, WorkOrderHistory, WorkOrderLineItemChangeEvent,
 WorkOrderLineItemHistory, WorkOrderLineItemStatus, WorkOrderShare, WorkOrderStatus,
 WorkPlanChangeEvent, WorkPlanHistory, WorkPlanShare, WorkPlanTemplateChangeEvent,
 WorkPlanTemplateEntryChangeEvent, WorkPlanTemplateEntryHistory,
 WorkPlanTemplateHistory, WorkPlanTemplateShare, WorkStepChangeEvent, WorkStepHistory,
 WorkStepStatus, WorkStepTemplateChangeEvent, WorkStepTemplateHistory,
 WorkStepTemplateShare, WorkThanksShare
```

Nicht unterstützte Entitäten und Felder für ServiceNow

Die folgenden ServiceNow Entitäten oder Felder werden für die Verwendung in einer Zero-ETL-Integration mit einer Quelle nicht unterstützt. ServiceNow

```
ais_acl_overrides, ais_async_genius_result, ais_async_request, ais_connection,
 ais_genius_result_configuration_parameters, ais_partition_health,
 ais_partition_health_response, ais_publish_history, ais_relevancy_training_execution,
 ais_relevancy_training_staging, ais_search_profile_relevancy_model,
 catalog_draft_entities, clone_log, clone_log0000, clone_log0001, clone_log0002,
 clone_log0003, clone_log0004, clone_log0005, clone_log0006, clone_log0007,
 cmdb_ie_context, cmdb_ie_log, cmdb_ie_run, cmdb_ire_partial_payloads_index,
 cmdb_qb_result_base$par1, cmdb$par1, discovery_log, discovery_log0000,
 discovery_log0001, discovery_log0002, discovery_log0003, discovery_log0004,
 discovery_log0005, discovery_log0006, discovery_log0007, ecc_agent_log,
 entitlement_data, entl_subscription_map, gs_entitlement_plugin_mapping,
 ih_transaction_exclusion, import_log, import_log0000, import_log0001, import_log0002,
 import_log0003, import_log0004, import_log0005, import_log0006, import_log0007,
 jrobin_archive, jrobin_database, jrobin_datasource, jrobin_definition, jrobin_graph,
 jrobin_graph_line, jrobin_graph_set, jrobin_graph_set_member, jrobin_shard,
 jrobin_shard_location, license_role_discovery_run, logger_configuration_validation,
 m2m_analytics_event_logger, m2m_user_consent_info, ml_artifact_object_store,
 np$sys_gen_ai_filter_sample, np$sys_ui_element, np$sys_ui_list_element,
 one_api_service_plan_feature_invocation, one_api_service_plan_invocation,
 open_nlu_predict_log, open_nlu_predict_log0000, open_nlu_predict_log0001,
 open_nlu_predict_log0002, open_nlu_predict_log0003, open_nlu_predict_log0004,
 open_nlu_predict_log0005, open_nlu_predict_log0006, open_nlu_predict_log0007,
 pa_diagnostic_log, pa_diagnostic_log0000, pa_diagnostic_log0001,
 pa_diagnostic_log0002, pa_diagnostic_log0003, pa_diagnostic_log0004,
 pa_diagnostic_log0005, pa_diagnostic_log0006, pa_diagnostic_log0007, pa_favorites,
 pa_job_log_rows, pa_job_log_rows0000, pa_job_log_rows0001, pa_job_log_rows0002,
 pa_job_log_rows0003, pa_job_log_rows0004, pa_job_log_rows0005, pa_job_log_rows0006,
 pa_job_log_rows0007, pa_migration_ignored_scores, pa_scores_l1, pa_scores_l2,
```

```
pa_scores_migration_groups, par_dashboard_conversion_backup, promin_log,
promin_log0000, promin_log0001, promin_log0002, promin_log0003, promin_log0004,
promin_log0005, promin_log0006, promin_log0007, promin_request_object,
proposed_change_verification_log, proposed_change_verification_log0000,
proposed_change_verification_log0001, proposed_change_verification_log0002,
proposed_change_verification_log0003, proposed_change_verification_log0004,
proposed_change_verification_log0005, proposed_change_verification_log0006,
proposed_change_verification_log0007, protected_table_log, protected_table_log0000,
protected_table_log0001, protected_table_log0002, protected_table_log0003,
protected_table_log0004, protected_table_log0005, protected_table_log0006,
protected_table_log0007, pwd_history, qb_query_results, scan_log,
scan_log0000, scan_log0001, scan_log0002, scan_log0003, scan_log0004,
scan_log0005, scan_log0006, scan_log0007, schema_validator_error,
sla_repair_log_entry, sla_repair_log_entry0000, sla_repair_log_entry0001,
sla_repair_log_entry0002, sla_repair_log_entry0003, sla_repair_log_entry0004,
sla_repair_log_entry0005, sla_repair_log_entry0006, sla_repair_log_entry0007,
sla_repair_log_message, sla_repair_log_message0000, sla_repair_log_message0001,
sla_repair_log_message0002, sla_repair_log_message0003, sla_repair_log_message0004,
sla_repair_log_message0005, sla_repair_log_message0006, sla_repair_log_message0007,
sn_bm_client_activity, sn_ci_analytics_st_actionable_notifs,
sn_ci_analytics_st_conv_completion_by_cat, sn_ci_analytics_st_conv_dynamic_property,
sn_ci_analytics_st_conversation, sn_ci_analytics_st_count_by_date,
sn_ci_analytics_st_event_occurrence, sn_ci_analytics_st_event_property_value_trend,
sn_ci_analytics_st_issue_auto_resolution, sn_ci_analytics_st_no_clicks,
sn_ci_analytics_st_no_results, sn_ci_analytics_st_property_summary_by_event,
sn_ci_analytics_st_session_count_per_locale, sn_ci_analytics_st_session_duration,
sn_ci_analytics_st_spokes_usage, sn_ci_analytics_st_topic_execution_stats,
sn_ci_analytics_st_topic_occurrence, sn_ci_analytics_st_trending_content,
sn_ci_analytics_st_trending_queries, sn_ci_analytics_st_users,
sn_cs_plugin_signatures, sn_cs_telemetry_log, sn_dfc_application, sn_dfc_product,
sn_employee_position, sn_entitlement_st_subscription_application_family,
sn_entitlement_st_subscription_application_users, sn_hr_sp_st_relevant_for_you,
sn_instance_clone_log, sn_instance_clone_log0000, sn_instance_clone_log0001,
sn_instance_clone_log0002, sn_instance_clone_log0003, sn_instance_clone_log0004,
sn_instance_clone_log0005, sn_instance_clone_log0006, sn_instance_clone_log0007,
sn_km_mr_st_kb_knowledge, sn_me_st_topic, sn_rf_conditional_definition,
sn_rf_evaluation_type, sn_rf_evaluation_type_input, sn_rf_recommendation_action,
sn_rf_recommendation_experience, sn_rf_recommendation_history,
sn_rf_recommendation_rule, sn_rf_record_display_configuration, sn_rf_trend_definition,
sn_sub_man_st_account_level_entitlement, sn_sub_man_st_gen_ai_metadata,
sn_sub_man_st_instance_used_assist_count, sn_sub_man_st_now_assist_creator_instances,
sn_sub_man_st_now_assists_aggregate, sn_sub_man_st_subscribed_groups,
sn_sub_man_st_subscription_insights, sn_sub_man_st_subscription_license_detail_metric,
sn_sub_man_st_unallocated_group_recommendation, sn_sub_man_st_unconfirmed_user_group,
```

```
sn_wn_user_app_activity, sn_wn_user_content_activity, snc_monitorable_item,
snpar_sched_export_v_scheduled_export_visualization, spotlight,
spotlight_audit, spotlight_copy_log_row, spotlight_copy_log_row0000,
spotlight_copy_log_row0001, spotlight_copy_log_row0002, spotlight_copy_log_row0003,
spotlight_copy_log_row0004, spotlight_copy_log_row0005, spotlight_copy_log_row0006,
spotlight_copy_log_row0007, spotlight_job_log_row, spotlight_job_log_row0000,
spotlight_job_log_row0001, spotlight_job_log_row0002, spotlight_job_log_row0003,
spotlight_job_log_row0004, spotlight_job_log_row0005, spotlight_job_log_row0006,
spotlight_job_log_row0007, st_dfc_performance_metric, st_license_detail_metric,
st_on_call_hour, st_sc_wizard_question, st_sys_catalog_items_and_variable_sets,
st_sys_design_system_icon, subscription_instance_stats, svc_container_config,
svc_environment_config, svc_layer_config, svc_model_assoc_ci,
svc_model_checkpoint_attr, svc_model_obj_cluster, svc_model_obj_constraint,
svc_model_obj_deployable, svc_model_obj_element, svc_model_obj_impact,
svc_model_obj_impactrule, svc_model_obj_package, svc_model_obj_path,
svc_model_obj_relation, svc_model_obj_service, sys_administrative_script_transaction,
sys_amb_message, sys_amb_processor, sys_analytics_batch_state,
sys_analytics_config, sys_analytics_data_points_error, sys_analytics_event,
sys_analytics_logger, sys_analytics_logger_field, sys_app_payload_loader_rule,
sys_app_payload_unloader_rule, sys_app_scan_payload, sys_app_scan_variable,
sys_app_scan_variable_type, sys_archive_destroy_log, sys_archive_destroy_run,
sys_archive_log, sys_archive_run, sys_atf_transaction_log, sys_attachment_doc,
sys_attachment_doc_v2, sys_attachment_soft_deleted, sys_audit, sys_audit_relation,
sys_auth_policy_api_allowed, sys_aw_registered_scripting_modal,
sys_cache_flush, sys_data_egress_source, sys_dm_delete_count,
sys_export_set_log, sys_export_set_log0000, sys_export_set_log0001,
sys_export_set_log0002, sys_export_set_log0003, sys_export_set_log0004,
sys_export_set_log0005, sys_export_set_log0006, sys_export_set_log0007,
sys_flow_compiled_flow, sys_flow_compiled_flow_chunk, sys_flow_context_chunk,
sys_flow_context_chunk_archive, sys_flow_context_inputs_chunk,
sys_flow_execution_history, sys_flow_log, sys_flow_log0000, sys_flow_log0001,
sys_flow_log0002, sys_flow_log0003, sys_flow_plan_context_binding,
sys_flow_report_doc, sys_flow_report_doc_chunk, sys_flow_report_doc_chunk_archive,
sys_flow_runtime_state_chunk, sys_flow_runtime_value_chunk,
sys_flow_subflow_plan_chunk, sys_flow_trigger_plan_chunk, sys_flow_val_listener,
sys_flow_value, sys_flow_value_chunk, sys_gen_ai_config_example,
sys_gen_ai_feature_mapping, sys_gen_ai_strategy_mapping, sys_gen_ai_usage_log,
sys_generative_ai_capability_definition, sys_generative_ai_log,
sys_generative_ai_response_validator, sys_generative_ai_validator,
sys_geo_routing, sys_geo_routing_config, sys_hop_token, sys_hub_action_plan_chunk,
sys_hub_snapshot_chunk, sys_journal_field, sys_journal_field_edit, sys_json_chunk,
sys_kaa_policy, sys_kaa_subidentity_assertion, sys_kaa_user_policy_mapping,
sys_mapplication, sys_mass_encryption_job, sys_notification_execution_log,
sys_notification_execution_log0000, sys_notification_execution_log0001,
```

```
sys_notification_execution_log0002, sys_notification_execution_log0003,
sys_notification_execution_log0004, sys_notification_execution_log0005,
sys_notification_execution_log0006, sys_notification_execution_log0007,
sys_nowmq_message, sys_nowmq_provider_param_definition, sys_orchestrator_action,
sys_pd_asset_configuration, sys_pd_context_chunk, sys_pd_context_log,
sys_pd_snapshot_chunk, sys_pd_trigger_license, sys_processing_framework_job,
sys_query_index_hint, sys_query_rewrite, sys_query_string_log,
sys_replication_queue, sys_replication_queue0, sys_replication_queue1,
sys_replication_queue2, sys_replication_queue3, sys_replication_queue4,
sys_replication_queue5, sys_replication_queue6, sys_replication_queue7,
sys_request_performance, sys_rollback_blacklisted, sys_rollback_conflict,
sys_rollback_incremental, sys_rollback_log, sys_rollback_log0000,
sys_rollback_log0001, sys_rollback_log0002, sys_rollback_log0003,
sys_rollback_log0004, sys_rollback_log0005, sys_rollback_log0006,
sys_rollback_log0007, sys_rollback_run, sys_rollback_schema_change,
sys_rollback_schema_conflict, sys_rollback_sequence, sys_scheduler_assignment,
sys_scheduler_memory_pressure_job_log, sys_script_adapter_rule,
sys_script_batch_adapter_rule, sys_search_source_filter, sys_service_authentication,
sys_signing_job, sys_suggestion_reader, sys_sync_history_review, sys_trend,
sys_unreferenced_preview, sys_unreferenced_record_rule, sys_upgrade_manifest,
sys_upgrade_state, sys_ux_asset_cache_buster, sys_ux_lib_component_prop,
sys_ux_lib_presource, sys_ux_page_action, sys_ux_page_action_binding,
sysevent_queue_runtime, syslog, syslog_app_scope0000, syslog_app_scope0001,
syslog_app_scope0002, syslog_app_scope0003, syslog_app_scope0004,
syslog_app_scope0005, syslog_app_scope0006, syslog_app_scope0007, syslog_email0000,
syslog_email0001, syslog_email0002, syslog_email0003, syslog_email0004,
syslog_email0005, syslog_email0006, syslog_email0007, syslog_transaction,
syslog_transaction0000, syslog_transaction0001, syslog_transaction0002,
syslog_transaction0003, syslog_transaction0004, syslog_transaction0005,
syslog_transaction0006, syslog_transaction0007, syslog0000, syslog0001, syslog0002,
syslog0003, syslog0004, syslog0005, syslog0006, syslog0007, ts_attachment,
ts_c_1_0, ts_c_1_1, ts_c_1_2, ts_c_1_3, ts_c_1_4, ts_c_1_5, ts_c_1_6, ts_c_1_7,
ts_c_1_8, ts_c_1_9, ts_c_10_0, ts_c_10_1, ts_c_10_2, ts_c_10_3, ts_c_10_4,
ts_c_10_5, ts_c_10_6, ts_c_10_7, ts_c_10_8, ts_c_10_9, ts_c_11_0, ts_c_11_1,
ts_c_11_2, ts_c_11_3, ts_c_11_4, ts_c_11_5, ts_c_11_6, ts_c_11_7, ts_c_11_8,
ts_c_11_9, ts_c_2_0, ts_c_2_1, ts_c_2_2, ts_c_2_3, ts_c_2_4, ts_c_2_5, ts_c_2_6,
ts_c_2_7, ts_c_2_8, ts_c_2_9, ts_c_3_0, ts_c_3_1, ts_c_3_2, ts_c_3_3, ts_c_3_4,
ts_c_3_5, ts_c_3_6, ts_c_3_7, ts_c_3_8, ts_c_3_9, ts_c_39_0, ts_c_39_1, ts_c_39_2,
ts_c_39_3, ts_c_39_4, ts_c_39_5, ts_c_39_6, ts_c_39_7, ts_c_39_8, ts_c_39_9,
ts_c_40_0, ts_c_40_1, ts_c_40_2, ts_c_40_3, ts_c_40_4, ts_c_40_5, ts_c_40_6,
ts_c_40_7, ts_c_40_8, ts_c_40_9, ts_c_41_0, ts_c_41_1, ts_c_41_2, ts_c_41_3,
ts_c_41_4, ts_c_41_5, ts_c_41_6, ts_c_41_7, ts_c_41_8, ts_c_41_9, ts_c_42_0,
ts_c_42_1, ts_c_42_2, ts_c_42_3, ts_c_42_4, ts_c_42_5, ts_c_42_6, ts_c_42_7,
ts_c_42_8, ts_c_42_9, ts_c_43_0, ts_c_43_1, ts_c_43_2, ts_c_43_3, ts_c_43_4,
```

```
ts_c_43_5, ts_c_43_6, ts_c_43_7, ts_c_43_8, ts_c_43_9, ts_c_44_0, ts_c_44_1,
ts_c_44_2, ts_c_44_3, ts_c_44_4, ts_c_44_5, ts_c_44_6, ts_c_44_7, ts_c_44_8,
ts_c_44_9, ts_c_45_0, ts_c_45_1, ts_c_45_2, ts_c_45_3, ts_c_45_4, ts_c_45_5,
ts_c_45_6, ts_c_45_7, ts_c_45_8, ts_c_45_9, ts_c_6_0, ts_c_6_1, ts_c_6_2,
ts_c_6_3, ts_c_6_4, ts_c_6_5, ts_c_6_6, ts_c_6_7, ts_c_6_8, ts_c_6_9, ts_c_7_0,
ts_c_7_1, ts_c_7_2, ts_c_7_3, ts_c_7_4, ts_c_7_5, ts_c_7_6, ts_c_7_7, ts_c_7_8,
ts_c_7_9, ts_c_8_0, ts_c_8_1, ts_c_8_2, ts_c_8_3, ts_c_8_4, ts_c_8_5, ts_c_8_6,
ts_c_8_7, ts_c_8_8, ts_c_8_9, ts_c_attachment, ts_chain, ts_deleted_doc,
ts_document, ts_field, ts_index_stats, ts_phrase, ts_search_stats, ts_v4_attachment,
ts_word, ua_app_metadata, ua_audit_stats, ua_extra_page, ua_monitor_property,
ua_monitor_property_audit, ua_shared_service, ua_sn_table_inventory, ua_sp_known_bot,
ua_upload_log, v_ais_result_improvement_rule_condition_builder_values,
v_cluster_nodes, v_cxs_search_resource, v_db_index, v_db_trigger, v_file_load_order,
v_interaction_context, v_iostats, v_mysql_proclist, v_mysql_status, v_mysql_variables,
v_on_call_report_cache, v_pa_par_combined_dashboard, v_pd_activity_condition_to_run,
v_pd_activity_start_rule_with_condition, v_pd_lane_condition_to_run,
v_sql_debug, v_st_kb_category, v_st_kb_most_viewed, v_st_kb_recently_viewed,
v_st_km_genai_mra_similar_task, v_st_popular_item, v_st_recent_item, v_st_sc_cat_item,
v_st_sc_catalog, v_st_sc_category, validator_run_summary, vtb_card_history,
wf_log, wf_log0000, wf_log0001, wf_log0002, wf_log0003, wf_log0004, wf_log0005,
wf_log0006, wf_log0007, spotlight_criteria, pa_snapshots, pa_widget_indicators,
pa_widgets, sn_cim_register, global, gsw_change_log, gsw_content, gsw_content_group,
gsw_content_information, gsw_status_of_content, multi_factor_browser_fingerprint,
multi_factor_criteria, pa_filters, password_policy, plan_execution,
plan_mysql, plan_oracle, plan_postgres, sc_rest_api_without_access_policy,
sn_actsub_activity, sn_actsub_activity_fanout, sn_actsub_activity_stream,
sn_actsub_activity_type, sn_actsub_atype_attributes, sn_actsub_atype_notif_pref,
sn_actsub_module, sn_actsub_notif_object, sn_actsub_subobject_stream,
sn_actsub_subscribable_object, sn_actsub_subscription_notif_pref,
sn_actsub_user_stream, sn_appclient_store_outbound_http_quota,
sn_appcreator_app_template, sn_critical_update, sn_docker_spoke_images,
sn_employee_app, sn_employee_app_access, sn_employee_app_access_criteria,
sn_entitlement_genai_assist_counts, sn_entitlement_genai_creator_user_counts,
sn_entitlement_genai_creator_users, sn_mif_instance, sn_mif_sync_data,
sn_mif_sync_status, sn_mif_table_registration, sn_mif_trust_config,
sn_vsc_best_practice_configurations, sn_vsc_best_practice_goals,
sn_vsc_changed_hardening_settings, sn_vsc_changed_scan_findings,
sn_vsc_check_security_area, sn_vsc_elevation_event, sn_vsc_event,
sn_vsc_export_event, sn_vsc_export_setting, sn_vsc_harc_compliance_status_lookup,
sn_vsc_hardening_compliance_scores, sn_vsc_impersonation_event,
sn_vsc_instance_hardening_settings, sn_vsc_login_event,
sn_vsc_scan_comparisons, sn_vsc_scan_summary, sn_vsc_security_check_categories,
sn_vsc_security_check_configurations, sn_vsc_security_configuration_groups,
sn_vsc_security_privacy_capabilities, sn_vsc_updated_settings,
```

```
sn_vsc_user_comparisons, sys_app_hash_inventory, sys_coalesce_strategy_deferred,
sys_flow_secure_data, sys_formula_function, sys_geocoding_request,
sys_global_file_hash, sys_import_set_row, sys_index, sys_index_explain,
sys_installation_schedule, sys_installation_schedule_item, sys_offline_app,
sys_package, sys_package_dependency_item, sys_package_dependency_m2m,
sys_plugins, sys_querystat, sys_reap_package, sys_scoped_plugin,
sys_stage_storage_alias, sys_storage_alias, sys_storage_table_alias, sys_store_app,
sys_table_partition, sys_upgrade_history_log, sys_user_public_credential,
sys_webauthn_authentication_request, sys_webauthn_registration_request,
syslog_app_scope, syslog_email, syslog_page_timing, ua_instance_state_config,
v_expression_cache, v_private_cache, v_shared_cache, sys_amb_message0002,
sys_amb_message0004, sys_amb_message0005, sys_metadata, v_par_unified_report_viz
```

Konfiguration eines Zero-ETL-Integrationsziels

AWS Bei der Konfiguration eines Ziels für eine Zero-ETL-Integration stehen mehrere Optionen zur Verfügung. Das Ziel kann ein verschlüsseltes Amazon Redshift Data Warehouse oder ein Amazon SageMaker Lakehouse-Katalog sein.

Bevor Sie das Ziel für die Zero-ETL-Integration auswählen, müssen Sie eine der folgenden Zielressourcen konfigurieren.

Zu den Konfigurationsoptionen für ein Ziel in einer Zero-ETL-Integration gehören:

- Ein Amazon SageMaker Lakehouse-Katalog und eine Datenbank, die mit regulärem Amazon S3 S3-Speicher konfiguriert sind. Siehe Konfiguration eines Amazon SageMaker Lakehouse-Katalogs mit regulärem S3-Speicher.
- Ein Amazon SageMaker Lakehouse-Katalog, der mit dem Amazon S3 Tables-Bucket konfiguriert ist. Siehe Konfiguration von Amazon S3 S3-Tabellen als Ziel.
- Ein Amazon SageMaker Lakehouse-Katalog, der mit Amazon Redshift verwaltetem Speicher konfiguriert ist. Siehe Konfiguration eines Amazon SageMaker Lakehouse-Katalogs mit Amazon Redshift verwaltetem Speicher.
- Ein Amazon Redshift Redshift-Data Warehouse, das durch einen Redshift-Namespace identifiziert wird. Siehe Konfiguration eines Amazon Redshift Data Warehouse-Ziels.

> **Note**
>
> Sie können das Ziel einer Zero-ETL-Integration nach der Erstellung nicht ändern.

Konfiguration eines Amazon SageMaker Lakehouse-Katalogs mit regulärem S3-Speicher

In diesem Abschnitt werden die Voraussetzungen und Einrichtungsschritte für die Konfiguration eines regulären Amazon S3 S3-Buckets als Speicher für Ihr Amazon SageMaker Lakehouse-Katalogziel in einer Zero-ETL-Integration beschrieben.

Voraussetzungen für die Einrichtung einer Integration

Bevor Sie eine Zero-ETL-Integration mit einem Amazon SageMaker Lakehouse-Katalog unter Verwendung von normalem S3-Speicher erstellen, müssen Sie die folgenden Einrichtungsaufgaben ausführen:

1. Richten Sie eine Datenbank ein AWS Glue
2. Geben Sie die RBAC-Richtlinie für den Katalog an
3. Erstellen Sie eine IAM-Zielrolle

Nachdem Sie den Amazon SageMaker Lakehouse-Katalog mit regulärem Amazon S3 S3-Speicher konfiguriert haben, können Sie mit Konfiguration der Integration mit Ihrem Ziel dem Abschluss der Integrationseinrichtung fortfahren.

Konfiguration von Amazon S3 S3-Tabellen als Ziel

In diesem Abschnitt werden die Voraussetzungen und Einrichtungsschritte für die Konfiguration von Amazon S3 S3-Tabellen als Ziel für Ihre Zero-ETL-Integration beschrieben.

Voraussetzungen für die Einrichtung einer Integration

Bevor Sie eine Zero-ETL-Integration mit Amazon S3 S3-Tabellen als Ziel erstellen, müssen Sie die folgenden Einrichtungsaufgaben ausführen:

1. Bucket für Amazon S3 S3-Tabellen einrichten
2. Geben Sie die RBAC-Richtlinie für den Katalog an
3. Erstellen Sie eine IAM-Zielrolle

Bucket für Amazon S3 S3-Tabellen einrichten

1. Erstellen Sie einen S3-Tabellen-Bucket in Ihrem Konto, indem Sie den Anweisungen unter Erste Schritte mit Amazon S3 S3-Tabellen folgen.
2. Aktivieren Sie Analytics-Integrationen mit Ihrem S3-Table-Bucket, indem Sie diese Anweisungen befolgen: AWS Services in Amazon S3 S3-Tabellen integrieren.

Geben Sie die RBAC-Richtlinie für den Katalog an

Die folgenden Berechtigungen müssen der Catalog RBAC Policy hinzugefügt werden, um Integrationen zwischen Quell- und Amazon S3 S3-Tabellen-Katalogziel zu ermöglichen.

Die Ressourcenrichtlinie für den AWS Glue Zielkatalog muss Glue Service-Berechtigungen für enthalten AuthorizeInboundIntegration. Darüber hinaus sind CreateInboundIntegration Berechtigungen entweder für den Quellprinzipal, der die Integration erstellt, oder für die AWS Glue Zielressourcenrichtlinie erforderlich.

> **Note**
>
> Für ein kontenübergreifendes Szenario müssen sowohl die Quellprinzipal- als auch die AWS Glue Zielkatalog-Ressourcenrichtlinie Glue: CreateInboundIntegration -Berechtigungen für die Ressource enthalten.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
    // Optional for same account but mandatory for cross account scenarios
    // Allow Alice to create Integration on Target Catalog
      "Principal": {
        "AWS": [
          "arn:aws:iam::<source-account-id>:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:CreateInboundIntegration"
      ],
      "Resource": [
```

```
        "arn:aws:glue:<region>:<Target-Account-Id>:catalog/<s3tablescatalog>/*"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceArn": "arn:aws:dynamodb:<region>:<Account>:table/<table-name>"
        }
      }
    },
    { // Required: Allow Glue to Authorize the Inbound Integration on behalf of Bob
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:AuthorizeInboundIntegration"
      ],
      "Resource": [
        "arn:aws:glue:<region>:<Target-Account-Id>:catalog/<s3tablescatalog>/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "arn:aws:dynamodb:<region>:<account-id>:table/<table-name>"
        }
      }
    }
  ]
}
```

> **Note**
>
> `<s3tablescatalog>`Ersetzen Sie es durch den Katalognamen Ihrer S3-Tabellen.

Erstellen Sie eine IAM-Zielrolle

Erstellen Sie eine IAM-Zielrolle mit den folgenden Berechtigungen und Vertrauensbeziehungen:

Beispiel für eine IAM-Richtlinie:

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Action": [
                "s3tables:ListTableBuckets",
                "s3tables:GetTableBucket",
                "s3tables:GetTableBucketEncryption",
                "s3tables:GetNamespace",
                "s3tables:CreateNamespace",
                "s3tables:ListNamespaces",
                "s3tables:CreateTable",
                "s3tables:GetTable",
                "s3tables:GetTableEncryption",
                "s3tables:ListTables",
                "s3tables:GetTableMetadataLocation",
                "s3tables:UpdateTableMetadataLocation",
                "s3tables:GetTableData",
                "s3tables:PutTableData"
            ],
            "Resource": "arn:aws:s3tables:<region>:<account-id>:bucket/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/Glue/ZeroETL"
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
```

```
}
```

Fügen Sie der Target-IAM-Rolle die folgende Vertrauensrichtlinie hinzu, damit der AWS Glue Service sie übernehmen kann:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

> **Note**
>
> Stellen Sie sicher, dass es in der S3-Tables-Bucket-Ressourcenrichtlinie keine explizite DENY-Anweisung für diese Ziel-IAM-Rolle gibt. Eine explizite DENY-Anweisung würde alle ALLOW-Berechtigungen außer Kraft setzen und verhindern, dass die Integration ordnungsgemäß funktioniert.

Konfiguration eines Amazon SageMaker Lakehouse-Katalogs mit Amazon Redshift verwaltetem Speicher

In diesem Abschnitt werden die Voraussetzungen und Einrichtungsschritte für die Konfiguration eines Amazon SageMaker Lakehouse-Katalogs mit Amazon Redshift verwaltetem Speicher (RMS) als Ziel für Ihre Zero-ETL-Integration beschrieben.

Voraussetzungen für die Einrichtung einer Integration

Bevor Sie eine Zero-ETL-Integration mit einem Amazon SageMaker Lakehouse-Katalog mithilfe von Redshift Managed Storage erstellen, müssen Sie die folgenden Einrichtungsaufgaben ausführen:

1. Richten Sie einen Cluster oder eine serverlose Arbeitsgruppe ein Amazon Redshift
2. Registrieren Sie die Amazon Redshift Integration mit Lake Formation

3. Erstellen Sie einen verwalteten Katalog in Lake Formation
4. Konfigurieren Sie IAM-Berechtigungen

Amazon Redshift Verwalteten Speicher einrichten

So richten Sie Amazon Redshift verwalteten Speicher für Ihre Zero-ETL-Integration ein:

1. Erstellen oder verwenden Sie einen vorhandenen Amazon Redshift Redshift-Cluster oder eine serverlose Arbeitsgruppe. Stellen Sie sicher, dass in der Amazon Redshift Zielarbeitsgruppe oder dem Zielcluster der `enable_case_sensitive_identifier` Parameter aktiviert ist, damit die Integration erfolgreich ist. Weitere Informationen zur Aktivierung der Groß- und Kleinschreibung finden Sie unter Aktivieren der Groß- und Kleinschreibung für Ihr Data Warehouse im Amazon Redshift Redshift-Managementleitfaden.
2. Registrieren Sie eine Integration von Redshift in den Katalog unter. AWS Lake Formation Siehe Registrierung von Amazon Redshift Clustern und Namespaces im Datenkatalog. AWS Glue
3. Erstellen Sie einen föderierten oder verwalteten Katalog in. AWS Lake Formation Weitere Informationen finden Sie unter:
   - Amazon Redshift Daten in den AWS Glue Datenkatalog aufnehmen
   - Einen Amazon Redshift verwalteten Katalog im AWS Glue Datenkatalog erstellen
4. Konfigurieren Sie IAM-Berechtigungen für die Zielrolle. Die Rolle benötigt Berechtigungen für den Zugriff auf Redshift- und Lake Formation Formation-Ressourcen. Die Rolle sollte mindestens Folgendes haben:
   - Berechtigungen für den Zugriff auf den Redshift-Cluster oder die Redshift-Arbeitsgruppe
   - Berechtigungen für den Zugriff auf den Lake Formation-Katalog
   - Berechtigungen zum Erstellen und Verwalten von Tabellen im Katalog
   - CloudWatch und CloudWatch protokolliert Berechtigungen für die Überwachung

Nachdem Sie den Amazon SageMaker Lakehouse-Katalog mit verwaltetem Amazon Redshift Redshift-Speicher konfiguriert haben, können Sie mit Konfiguration der Integration mit Ihrem Ziel dem Abschluss der Integrationseinrichtung fortfahren.

Konfiguration eines Amazon Redshift Data Warehouse-Ziels

In diesem Abschnitt werden die Voraussetzungen und Einrichtungsschritte für die Konfiguration eines Amazon Redshift Data Warehouse als Ziel für Ihre Zero-ETL-Integration beschrieben.

Voraussetzungen für die Einrichtung einer Integration

Bevor Sie eine Zero-ETL-Integration mit einem Amazon Redshift Data Warehouse-Ziel erstellen, müssen Sie die folgenden Einrichtungsaufgaben ausführen:

1. Richten Sie einen Amazon Redshift Cluster oder eine serverlose Arbeitsgruppe ein
2. Konfigurieren Sie die Berücksichtigung von Groß-
3. Konfigurieren Sie IAM-Berechtigungen

Das Amazon Redshift Data Warehouse einrichten

So richten Sie ein Amazon Redshift Data Warehouse für Ihre Zero-ETL-Integration ein:

1. Navigieren Sie zur Amazon Redshift Konsole und klicken Sie auf Cluster erstellen oder verwenden Sie einen vorhandenen Cluster. Klicken Sie für Amazon Redshift Serverless auf Arbeitsgruppe erstellen.
2. Wenn Sie einen neuen Cluster erstellen, wählen Sie eine geeignete Clustergröße und stellen Sie sicher, dass Ihr Cluster verschlüsselt ist. Für Serverless konfigurieren Sie die Arbeitsgruppeneinstellungen entsprechend Ihren Anforderungen.
3. Stellen Sie sicher, dass in der Amazon Redshift Zielarbeitsgruppe oder dem Zielcluster der `enable_case_sensitive_identifier` Parameter aktiviert ist, damit die Integration erfolgreich ist. Weitere Informationen zur Aktivierung der Groß- und Kleinschreibung finden Sie unter Aktivieren der Groß- und Kleinschreibung für Ihr Data Warehouse im Amazon Redshift Redshift-Managementleitfaden.
4. Konfigurieren Sie IAM-Berechtigungen, damit die Zero-ETL-Integration auf Ihr Data Warehouse zugreifen kann. Amazon Redshift Sie müssen eine IAM-Rolle mit den folgenden Berechtigungen erstellen:
   - Berechtigungen für den Zugriff auf den Amazon Redshift Cluster oder die Arbeitsgruppe
   - Berechtigungen zum Erstellen und Verwalten von Datenbanken und Tabellen in Amazon Redshift
   - CloudWatch und Amazon CloudWatch protokolliert die Berechtigungen für die Überwachung
5. Nachdem die Amazon Redshift Arbeitsgruppen- oder Clustereinrichtung abgeschlossen ist, müssen Sie Ihr Data Warehouse für Zero-ETL-Integrationen konfigurieren. Weitere Informationen finden Sie unter Erste Schritte mit Zero-ETL-Integrationen im Amazon Redshift Management Guide.

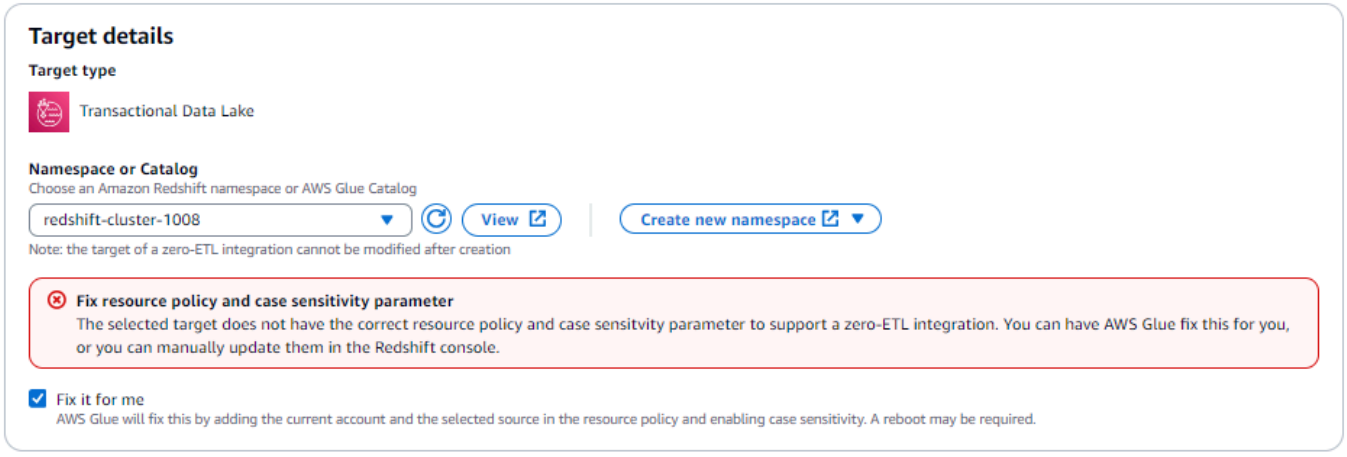
> **Note**
>
> Wenn Sie ein Amazon Redshift Data Warehouse als Ziel verwenden, erstellt die Integration ein Schema in der angegebenen Datenbank, um die replizierten Daten zu speichern. Der Schemaname wird vom Integrationsnamen abgeleitet.

Nach der Konfiguration des Amazon Redshift Data Warehouse können Sie mit Konfiguration der Integration mit Ihrem Ziel dem Abschluss der Integrationseinrichtung fortfahren.

Konfiguration der Integration mit Ihrem Ziel

Nachdem Sie Ihre Zielressourcen konfiguriert, Ihre Verbindung ausgewählt und eine Quell-IAM-Rolle angegeben haben, gehen Sie wie folgt vor, um die Einrichtung der Integration abzuschließen:

1. Geben Sie das Ziel an, das Sie in den vorherigen Schritten konfiguriert haben.
2. Wählen Sie die Option AWS Glue Repariere es für mich aus. Für das Amazon Redshift Ziel wird dies:
   - Wenden Sie einen autorisierten Dienstprinzipal auf den Amazon Redshift Cluster oder die serverlose Arbeitsgruppe an.
   - Wenden Sie einen autorisierten AWS Glue Quell-ARN auf den Amazon Redshift Cluster oder die serverlose Arbeitsgruppe an.
   - Ordnen Sie eine neue Parametergruppe zu. `enable_case_sensitive_identifier = true`



3. Geben Sie den Namen der Integration ein und wählen Sie „Integration erstellen und starten“.

4. Sobald sich Ihre Integration im aktiven Status befindet, navigieren Sie zur Seite mit den Integrationsdetails und wählen Sie Datenbank aus Integration erstellen aus.
5. Schließlich können Sie zum Redshift-Abfrage-Editor navigieren und eine Verbindung zu Ihrer Datenbank herstellen, um den Snapshot und die inkrementellen Daten zu validieren.

> **Note**
>
> Sie können im Namespace- oder Katalognamen nur alphanumerische Kleinbuchstaben und Unterstriche verwenden. Dies unterscheidet sich von dem, was der AWS Glue Datenkatalog ermöglicht, eine Datenbank mit einem beliebigen Namen (einschließlich Sonderzeichen) zu erstellen.

Leitfaden zur Partitionsspezifikation und Schemaentfernung

Bei der Arbeit mit NoSQL-Datenquellen wie DynamoDB und SaaS-Anwendungen stellen Daten oft besondere Herausforderungen für Analysen dar:

1. Datensätze in derselben Tabelle können ein anderes Schema haben
2. Verschachtelte Datensätze innerhalb derselben Tabelle können unterschiedlich dargestellt werden
3. Komplexe verschachtelte Strukturen wie Maps und Arrays müssen für effiziente Abfragen transformiert werden
4. Eine optimale Datenorganisation ist erforderlich, um eine skalierbare Abfrageleistung sicherzustellen

AWS Glue Zero-ETL-Integrationen lösen diese Herausforderungen mit zwei leistungsstarken Funktionen:

- Schema-Unnesting: Führt komplexe verschachtelte Datenstrukturen automatisch in analysefreundliche Formate zusammen. Dabei stehen konfigurierbare Ebenen zur Entfernung von Verschachtelungen zur Verfügung, um ein Gleichgewicht zwischen der Beibehaltung der Datenstruktur und der Optimierung zur Vereinfachung der Abfragen herzustellen.
- Datenpartitionierung: Organisiert Daten auf der Grundlage bestimmter Spalten oder zeitbasierter Dimensionen in logischen Partitionen. Dadurch wird die Abfrageleistung verbessert und die Kosten gesenkt, indem das Bereinigen von Partitionen während der Abfrageausführung aktiviert wird.

Um solche Datenquellen effektiv abzufragen, bietet AWS Glue Zero-ETL out-of-the-box Schemabehandlungs- und Partitionierungsschemata für Quelldaten, die in der Glue-Zieldatenbank repliziert werden. AWS Über die CreateIntegrationTableProperty API können Sie für jede Tabelle Einstellungen zum Entfernen von Verschachtelungen und Partitionierung von Schemas konfigurieren, sodass Sie genau steuern können, wie Daten für Analyse-Workloads strukturiert und organisiert werden.

Standardverhalten zum Entfernen von Verschachtelungen und Partitionierungen

1. AWS Glue Zero-ETL ist standardmäßig auf FULL Unnest gesetzt, wenn keine Unnesting-Optionen für die Zieltabelle bereitgestellt werden
2. AWS Glue Zero-ETL verwendet standardmäßig die Bucket-Partitionierung, wenn keine für die Zieltabelle angegeben PartitionSpec sind

Unverschachtelung von Schemas

Bei der Integration mit Analysediensten über Zero-ETL können Sie wählen, wie verschachtelte Strukturen in den Zieltabellen dargestellt werden. AWS Glue Zero-ETL bietet Optionen zum Entfernen von Schemas, um komplexe Datenstrukturen in analysefreundlichere Formate zu reduzieren.

Optionen zum Entfernen von Verschachtelungen

Wenn Sie eine Zero-ETL-Integration mit einer Quelle erstellen, können Sie aus den folgenden Optionen zum Entfernen von Verschachtelungen wählen. Diese Optionen entsprechen bestimmten Aufzählungswerten, die Sie beim Aufrufen der API verwenden werden. CreateIntegrationTableProperty

Keine Unverschachtelung (Standard)

API-Wert: **`NO_UNNEST`**

Behält die ursprüngliche verschachtelte Struktur der DynamoDB-Elemente bei. Karten und Listen werden als strukturierte Spalten im Ziel gespeichert.

Ideal für: Beibehaltung der exakten Struktur Ihrer DynamoDB-Daten, wenn Ihre Analysetools mit verschachtelten Daten arbeiten können.

Eine Ebene aufheben

API-Wert: **`TOP_LEVEL`**

Reduziert die oberste Ebene verschachtelter Maps in einzelne Spalten. Listenstrukturen bleiben verschachtelt.

Ideal für: Balance zwischen Erhaltung der Datenstruktur und Einfachheit der Abfrage, wenn Ihre DynamoDB-Elemente über ein konsistentes Schema verfügen.

Löscht alle Ebenen

API-Wert: **`FULL`**

Reduziert rekursiv alle verschachtelten Strukturen (Maps und Listen) in einzelne Spalten mit Punktnotation für die Benennung.

Ideal für: Maximierung der Einfachheit von Abfragen bei der Arbeit mit tief verschachtelten Strukturen und Analysetools, die flache Schemas bevorzugen.

> **Note**
>
> Eine vollständige Aufhebung der Verschachtelung kann zu sehr breiten Tabellen mit vielen Spalten führen, wenn Ihre DynamoDB-Daten variable oder tief verschachtelte Strukturen aufweisen.

Example Verwenden von Optionen zum Entfernen von Verschachtelungen in der API

Wenn Sie Schema-Unnesting über die CreateIntegrationTableProperty API konfigurieren, geben Sie die Option zum Entfernen von Verschachtelungen im Parameter an: UnnestSpec

```
aws glue create-integration-table-property
  --resource-arn "arn:aws:glue:us-east-1:123456789012:integration/my-integration"
  --table-name "my-table"
  --cli-input-json '{
      "TargetTableConfig": {
          "UnnestSpec": "FULL",
          "TargetTableName": "my-target-table",
      }
  }'
```

Beispiele zum Entfernen von Verschachtelungen

Stellen Sie sich ein DynamoDB-Element mit der folgenden Struktur vor:

```
{
  "ProductId": "P12345",
  "ProductDetails": {
    "Name": "Smartphone",
    "Brand": "TechCo",
    "Specifications": {
      "Color": "Black",
      "Storage": "128GB"
    }
  },
  "Reviews": [
    {
      "Rating": 5,
      "Comment": "Great product!"
    },
    {
      "Rating": 4,
      "Comment": "Good value."
    }
  ]
}
```

Kein Beispiel für das Entfernen von Verschachtelungen

Ohne Aufheben der Verschachtelung hätte die Zieltabelle folgende Spalten:

- ProductId (string)
- ProductDetails (Struktur)
- Rezensionen (Reihe von Strukturen)

Abfragen müssten Struktur- und Array-Zugriffsmuster verwenden:

```
SELECT
  ProductId,
  ProductDetails.Name,
  ProductDetails.Specifications.Color
FROM product_table;
```

Beispiel für Unnest auf einer Ebene

Ohne Verschachtelung auf einer Ebene hätte die Zieltabelle folgende Spalten:

- ProductId (string)
- ProductDetails_Name (Zeichenfolge)
- ProductDetails_Marke (Zeichenfolge)
- ProductDetails_Spezifikationen (Struktur)
- Rezensionen (Reihe von Strukturen)

Abfragen würden für die erste Ebene vereinfacht:

```
SELECT
  ProductId,
  ProductDetails_Name,
  ProductDetails_Specifications.Color
FROM product_table;
```

Beispiel für Unnest all Levels

Ohne Verschachtelung aller Ebenen hätte die Zieltabelle Spalten und Werte:

Spaltenname (Typ)	Wert
ProductId (string)	P12345
ProductDetails_Name (Zeichenfolge)	Smartphone
ProductDetails_Marke (Zeichenfolge)	TechCo
ProductDetails_Specifications_Color (Zeichenfolge)	Schwarz
ProductDetails_Specifications_Storage (Zeichenfolge)	128 GB
Reviews_0_Rating (Zahl)	5
Reviews_0_Comment (Zeichenfolge)	Tolles Produkt!

Spaltenname (Typ)	Wert
Reviews_1_Bewertung (Zahl)	4
Reviews_1_Comment (Zeichenfolge)	Guter Wert.

Abfragen würden vollständig vereinfacht:

```
SELECT
  ProductId,
  ProductDetails_Name,
  ProductDetails_Specifications_Color
FROM product_table;
```

Datenpartitionierung

Was ist Datenpartitionierung?

Datenpartitionierung ist eine Technik, die große Datenmengen in kleinere, besser verwaltbare Segmente unterteilt, die als Partitionen bezeichnet werden. Im Kontext von AWS Glue Zero-ETL-Integrationen organisiert die Partitionierung Ihre Daten am Zielort auf der Grundlage bestimmter Spaltenwerte oder Transformationen dieser Werte.

Vorteile der Datenpartitionierung

Eine effektive Datenpartitionierung bietet mehrere wichtige Vorteile für Analyse-Workloads:

- Verbesserte Abfrageleistung: Abfragen können irrelevante Partitionen überspringen (Partitionsbereinigung), wodurch die Datenmenge, die gescannt werden muss, reduziert wird.
- Geringere Kosten: Indem Sie weniger Daten scannen, können Sie den I/O Rechenaufwand und die Kosten für Ihre Analyseabfragen senken.
- Bessere Skalierbarkeit: Die Partitionierung ermöglicht die parallel Verarbeitung von Datensegmenten und ermöglicht so eine effizientere Skalierung von Analyse-Workloads.
- Vereinfachtes Datenlebenszyklusmanagement: Sie können Aufbewahrungsrichtlinien auf Partitionsebene verwalten, was das Archivieren oder Löschen älterer Daten erleichtert.

Die wichtigsten Konzepte der Partitionierung

Spalten partitionieren

Spalten in Ihren Daten, anhand derer bestimmt wird, wie Datensätze in Partitionen organisiert sind. Effektive Partitionsspalten sollten sich an gängigen Abfragemustern orientieren und eine angemessene Kardinalität aufweisen.

Funktionen der Partition

Transformationen, die auf Partitionsspaltenwerte angewendet wurden, um die tatsächlichen Partitionsgrenzen zu erstellen. Beispiele hierfür sind Identitäts- (unter Verwendung des Rohwerts) und zeitbasierte Funktionen (Jahr, Monat, Tag, Stunde).

Bereinigen von Partitionen

Der Prozess, bei dem die Abfrage-Engine Partitionen identifiziert und überspringt, die keine relevanten Daten für eine Abfrage enthalten, wodurch die Leistung erheblich verbessert wird.

Granularität der Partition

Der Detaillierungsgrad, mit dem Daten partitioniert werden. Eine feinere Granularität (mehr Partitionen) kann die Abfrageleistung verbessern, kann jedoch den Metadaten-Overhead erhöhen. Eine gröbere Granularität (weniger Partitionen) reduziert den Metadaten-Overhead, kann aber dazu führen, dass mehr Daten als nötig gescannt werden.

Partitionierung in AWS Glue Zero-ETL-Integrationen

AWS Glue Zero-ETL-Integrationen verwenden das Apache Iceberg-Tabellenformat, das erweiterte Partitionierungsfunktionen bietet. Wenn Sie eine Zero-ETL-Integration erstellen, können Sie:

- Verwenden Sie standardmäßige Partitionierungsstrategien, die für Ihre Datenquelle optimiert sind
- Definieren Sie benutzerdefinierte Partitionierungsspezifikationen, die auf Ihre Abfragemuster zugeschnitten sind
- Wenden Sie Transformationen auf Partitionsspalten an (besonders nützlich für zeitstempelbasierte Partitionierung)
- Kombinieren Sie mehrere Partitionsstrategien für die Partitionierung auf mehreren Ebenen

Partitionierungskonfigurationen werden bei der Einrichtung Ihrer Zero-ETL-Integration über die `CreateIntegrationTableProperty` API angegeben. Nach der Konfiguration wendet AWS Glue diese Partitionierungsstrategien automatisch an, um Ihre Daten am Zielort zu organisieren.

API-Referenz zur Partitionsspezifikation

Verwenden Sie die folgenden Parameter in der CreateIntegrationTableProperties API, um die Partitionierung zu konfigurieren:

PartitionSpec

Eine Reihe von Partitionsspezifikationen, die definieren, wie Daten am Zielort partitioniert werden.

```
{
  "partitionSpec": [
    {
      "fieldName": "timestamp_col",
      "functionSpec": "month",
      "conversionSpec": "epoch_milli"
    },
    {
      "fieldName": "category",
      "functionSpec": "identity"
    }
  ]
}
```

FieldName

Eine UTF-8-Zeichenfolge (1—128 Byte), die den Spaltennamen angibt, der für die Partitionierung verwendet werden soll.

FunctionSpec

Gibt die Partitionierungsfunktion an. Zulässige Werte:

- `identity`- Verwendet Quellwerte direkt ohne Transformation
- `year`- Extrahiert das Jahr aus Zeitstempelwerten (z. B. 2023)
- `month`- Extrahiert den Monat aus Zeitstempelwerten (z. B. 2023-01)
- `day`- Extrahiert den Tag aus Zeitstempelwerten (z. B. 2023-01-15)
- `hour`- Extrahiert die Stunde aus Zeitstempelwerten (z. B. 2023-01-15-14)

> **Note**
>
> Zeitbasierte Funktionen (`year`,,,`hour`) erfordern `monthday`, dass der `ConversionSpec` Parameter das Quell-Zeitstempelformat angibt.

ConversionSpec

Eine UTF-8-Zeichenfolge, die das Zeitstempelformat der Quelldaten angibt. Gültige Werte für sind:

- `epoch_sec`- Zeitstempel der Unix-Epoche in Sekunden
- `epoch_milli`- Zeitstempel der Unix-Epoche in Millisekunden
- `iso`- Zeitstempel im ISO 8601-Format

Strategien zur Partitionierung

Standard-Partitionierung

Wenn keine Partitionsspalten angegeben sind, wendet AWS Glue Zero-ETL standardmäßige Partitionierungsstrategien an, die für Ihre Datenquelle optimiert sind:

- Primärschlüsselbasierte Partitionierung: Bei Quellen mit Primärschlüsseln (wie DynamoDB-Tabellen) partitioniert AWS Glue Zero-ETL Daten automatisch mithilfe des Primärschlüssels mit Bucketing, um eine Explosion der Partitionen zu verhindern.

Die Standardpartitionierung ist so konzipiert, dass sie sich gut für gängige Abfragemuster eignet, ohne dass eine manuelle Konfiguration erforderlich ist. Für bestimmte Abfragemuster oder Leistungsanforderungen möchten Sie jedoch möglicherweise benutzerdefinierte Partitionierungsstrategien definieren.

Benutzerdefinierte Partitionierungsstrategien

AWS Mit Glue Zero-ETL können Sie mithilfe des Parameters benutzerdefinierte Partitionierungsstrategien definieren. `PartitionSpec` Sie können eine oder mehrere Partitionsspalten angeben und auf jede Spalte unterschiedliche Partitionierungsfunktionen anwenden.

Bei der Identitätspartitionierung werden die Rohwerte einer Spalte verwendet, um Partitionen zu erstellen. Diese Strategie ist nützlich für Spalten mit niedriger bis mittlerer Kardinalität, z. B. Kategorie-, Regions- oder Statusfelder.

Example Beispiel für die Partitionierung von Identitäten

```
{
  "partitionSpec": [
    {
      "fieldName": "category",
      "functionSpec": "identity"
    }
  ]
}
```

Dadurch werden separate Partitionen für jeden eindeutigen Wert in der Spalte „Kategorie“ erstellt.

> **Warning**
>
> Vermeiden Sie die Identitätspartitionierung mit Spalten mit hoher Kardinalität (wie Primärschlüssel oder Zeitstempel), da dies zu einer Explosion der Partitionen führen kann, was die Leistung beeinträchtigt und den Metadaten-Overhead erhöht.

Bei der zeitbasierten Partitionierung werden Daten auf der Grundlage von Zeitstempelwerten mit unterschiedlichen Granularitäten (Jahr, Monat, Tag oder Stunde) organisiert. Diese Strategie ist ideal für Zeitreihendaten und ermöglicht effiziente Zeitbereichsabfragen.

Bei Verwendung der zeitbasierten Partitionierung kann AWS Glue Zero-ETL verschiedene Zeitstempelformate automatisch in ein standardisiertes Format konvertieren, bevor die Partitionsfunktion angewendet wird. Diese Konvertierung wird mithilfe des Parameters angegeben. `ConversionSpec`

Example Beispiel für eine zeitbasierte Partitionierung

```
{
  "partitionSpec": [
    {
      "fieldName": "created_at",
      "functionSpec": "month",
```

```
      "conversionSpec": "epoch_milli"
    }
  ]
}
```

Dabei werden Daten auf der Grundlage der Spalte „created_at“, die Zeitstempel der Unix-Epoche in Millisekunden enthält, nach Monaten partitioniert.

AWS Glue Zero-ETL unterstützt die folgenden zeitbasierten Partitionsfunktionen:

- Jahr: Partitioniert Daten nach Jahren (z. B. 2023, 2024)
- Monat: Partitionsdaten nach Monaten (z. B. 2023-01, 2023-02)
- Tag: Partitioniert Daten nach Tagen (z. B. 2023-01-01, 2023-01-02)
- Stunde: Partitioniert Daten nach Stunden (z. B. 2023-01-01, 2023-01-01-02)

AWS Glue Zero-ETL unterstützt über den Parameter die folgenden Zeitstempelformate: `ConversionSpec`

- epoch_sec: Zeitstempel der Unix-Epoche in Sekunden
- epoch_milli: Zeitstempel der Unix-Epoche in Millisekunden
- iso: Zeitstempel im ISO-8601-Format

> **Note**
>
> Die ursprünglichen Spaltenwerte bleiben in Ihren Quelldaten unverändert. AWS Glue wandelt in der Zieldatenbanktabelle nur Partitionsspaltenwerte in den Timestamp-Typ um. Die Transformationen gelten nur für den Partitionierungsprozess.

Die mehrstufige Partitionierung kombiniert mehrere Partitionsstrategien, um ein hierarchisches Partitionierungsschema zu erstellen. Dies ist nützlich, um verschiedene Arten von Abfragen für denselben Datensatz zu optimieren.

Example Beispiel für eine Partitionierung auf mehreren Ebenen

```
{
  "partitionSpec": [
```

```
    {
      "fieldName": "created_at",
      "functionSpec": "month",
      "conversionSpec": "iso"
    },
    {
      "fieldName": "region",
      "functionSpec": "identity"
    }
  ]
}
```

Dadurch wird ein zweistufiges Partitionierungsschema erstellt: zuerst nach Monat (aus der Spalte „created_at“), dann nach Region. Dies ermöglicht effiziente Abfragen, die nach Datumsbereichen, bestimmten Regionen oder einer Kombination dieser Dimensionen filtern.

Beim Entwerfen von Partitionierungsschemata mit mehreren Ebenen sollten Sie Folgendes berücksichtigen:

- Platzieren Sie Spalten mit höherer Selektivität an erster Stelle in der Partitionshierarchie
- Ausbalancieren der Partitionsgranularität mit der Anzahl der Partitionen
- Ausrichtung des Partitionierungsschemas an Ihren gängigsten Abfragemustern

Bewährte Methoden

Auswahl der Partitionsspalten

- Verwenden Sie bei der Partitionsfunktion keine Spalten mit hoher Kardinalität. `identity` Durch die Verwendung von Spalten mit hoher Kardinalität und Identitätspartitionierung entstehen viele kleine Partitionen, wodurch die Aufnahmeleistung erheblich beeinträchtigt werden kann. Spalten mit hoher Kardinalität können Folgendes beinhalten:
  - Primärschlüssel
  - Zeitstempelfelder (wie,) `LastModifiedTimestamp` `CreatedDate`
  - Vom System generierte Zeitstempel
- Wählen Sie nicht mehrere Timestamp-Partitionen in derselben Spalte aus. Zum Beispiel:

```
"partitionSpec": [
      {"fieldName": "col1", "functionSpec": "year", "conversionSpec" :
 "epoch_milli"},
```

```
      {"fieldName": "col1", "functionSpec": "month", "conversionSpec" :
 "epoch_milli"},
      {"fieldName": "col1", "functionSpec": "day", "conversionSpec" : "epoch_milli"},
      {"fieldName": "col1", "functionSpec": "hour", "conversionSpec" : "epoch_milli"}
]
```

Auswahl der Partition FunctionSpec/ConversionSpec

- Geben Sie das korrekte Format ConversionSpec (epoch_sec | epoch_milli | iso) an, das das Format der Spaltenwerte darstellt, die für die zeitstempelbasierte Partitionierung ausgewählt wurden, wenn Sie zeitstempelbasierte Partitionsfunktionen verwenden. AWS Glue Zero-ETL verwendet diesen Parameter, um Quelldaten vor der Partitionierung korrekt in das Zeitstempelformat umzuwandeln.
- Verwenden Sie die entsprechende Granularität (year/month/day/hour), die auf dem Datenvolumen basiert.
- Berücksichtigen Sie bei der Verwendung von ISO-Zeitstempeln die Auswirkungen auf die Zeitzone. AWS Glue Zero-ETL füllt alle Datensatzwerte der ausgewählten Zeitstempelspalte mit der UTC-Zeitzone.

Fehlerbehandlung

Status NEEDS_ATTENTION

Eine Integration wechselt in den Status NEEDS_ATTENTION, wenn:

- Die angegebenen Partitionsspalten sind in der Quelle nicht vorhanden
- Die Zeitstempelkonvertierung schlägt für Partitionsspalten fehl

Einschränkungen

Einschränkungen bei der Partitionierung

- Partitionsspezifikationen können nicht geändert werden, nachdem eine Integration erstellt wurde. Um eine andere Partitionierungsstrategie zu verwenden, müssen Sie eine neue Integration erstellen.
- Die maximale Anzahl von Partitionsspalten ist auf 10 begrenzt.

Einschränkungen bei der kontoübergreifenden Integration

- Bei der Erstellung kontoübergreifender Integrationen hat AWS Glue Console eine Einschränkung, da sie keine CreateIntegrationTableProperty API für die Konfiguration UnnestSpec und PartitionSpec für AWS Glue-Zieltabellen aufruft, die in dem Konto gehostet werden, für das keine Integration vorhanden ist.

  Problemumgehung: Die CreateIntegrationTableProperty API muss von CX von dem Konto aus aufgerufen werden, in dem die Zieldatenbank existiert.

Einschränkungen bei mehreren Integrationen

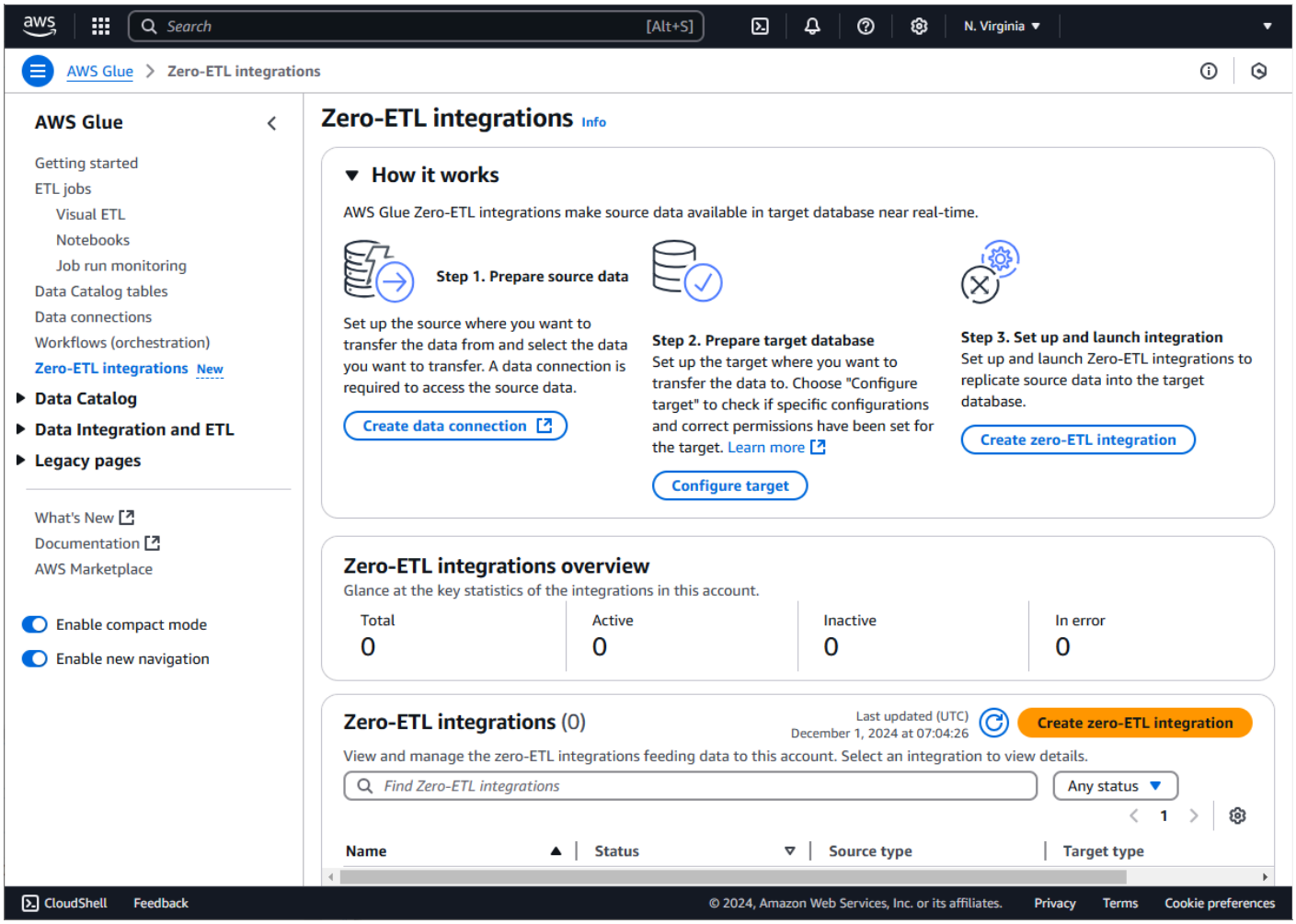
- Wenn Sie dieselbe Quelle mit unterschiedlichen unnest/partition Schemakonfigurationen replizieren müssen, müssen Sie für jede Integration separat eine neue AWS Glue-Datenbank erstellen. Rufen Sie später CreateIntegrationTableProperty für jede Tabelle aus einer individuellen AWS Glue-Datenbank mit den gewünschten Schema-Unverschachtelungs- und Partitionierungskonfigurationen auf.

Allgemeine Integrationsaufgaben

Eine Integration erstellen

In diesem Abschnitt werden die allgemeinen Schritte zum Erstellen einer Integration beschrieben. In diesem Beispiel wird Amazon DynamoDB als Quelle verwendet.

1. Wählen Sie auf der Startseite der AWS Glue Konsole Zero-ETL-Integrationen aus.
2. Sie können alle Ihre Integrationen auf der Zero ETL-Integrationshomepage einsehen. Um eine neue Integration zu erstellen, wählen Sie Zero-ETL-Integration erstellen aus.



3. Sie werden aufgefordert, einen Quelltyp auszuwählen. Wählen Sie Ihre Quelle aus und klicken Sie auf Weiter. Informationen zu SaaS-Integrationsquellen finden Sie in den Abschnitten zur Quellkonfiguration.
4. Wählen Sie auf der Seite Quelle und Ziel konfigurieren die Tabellen oder Entitäten aus, die repliziert werden sollen. Stellen Sie für Amazon DynamoDB sicher, dass die PITR- und RBAC-Richtlinie konfiguriert ist.
5. Geben Sie Ihr Integrationsziel an:
   - Wählen Sie für ein AWS Glue Datenkatalogziel die AWS Glue Datenbank aus, in die Sie die Daten replizieren möchten.
   - Wählen Sie für ein Amazon Redshift Data Warehouse-Ziel den Redshift-Cluster-Namespace oder den Redshift Serverless Workgroup-Namespace aus.

   Weitere Informationen finden Sie unter Konfiguration der Integration mit Ihrem Ziel.
6. Geben Sie die Ziel-IAM-Rolle an, die Sie in den Voraussetzungen erstellt haben.

7. Wenn Sie einen optionalen Ziel-KMS-Schlüssel für Ihre im Ziel gespeicherten Daten konfigurieren möchten, geben Sie einen aktivierten KMS-Schlüssel an. Wenn Sie eine Zielnetzwerkverbindung konfigurieren möchten, wählen Sie ebenfalls eine AWS Glue Verbindung aus.
8. Mit der Schaltfläche „Ziel korrigieren“ werden einige der Schritte im Abschnitt „Voraussetzungen“ dieser Dokumentation konfiguriert. Es stellt nämlich 1) eine Katalog-RBAC-Richtlinie bereit und 2) wenn kein Amazon S3 S3-URI angegeben wird, generiert es einen für Sie, andernfalls verwendet es den angegebenen URI.
9. Für Integrationen mit einem Redshift Data Warehouse-Ziel:
10. Wählen Sie auf der Seite Quelle und Ziel konfigurieren im Abschnitt Ausgabeeinstellungen die Option zum Entfernen von Schemaverschachtelungen aus, die Sie für Ihre Daten im Ziel verwenden möchten. Wenn Sie Kundenpartitionsschlüssel für Ihre Daten verwenden möchten, wählen Sie Benutzerdefinierte Partitionsschlüssel angeben aus und geben Sie bis zu 10 Schlüssel an. Andernfalls können Sie einfach die Partitionsschlüssel verwenden, die Ihrer zu replizierenden DynamoDB-Tabelle zugewiesen sind.
11. Im Abschnitt Sicherheit und Datenverschlüsselung können Sie einen KMS-Schlüssel angeben, der bei der Zwischenreplikation Ihrer Daten auf das Ziel verwendet wird. Andernfalls wird ein AWS verwalteter KMS-Schlüssel verwendet. Derzeit unterstützen wir nur eine 15-minütige Replikationseinstellung. Geben Sie in den Integrationsdetails einen Namen für die Zero-ETL-Integration ein.
12. Überprüfen Sie und stellen Sie sicher, dass alle angegebenen Details korrekt sind. Sobald alles bestätigt wurde, klicken Sie auf Integration erstellen und starten.
13. Auf der Zero ETL-Startseite können Sie die von Ihnen erstellte Integration auswählen und die Details für Ihre Integrationen werden angezeigt. Der „Status“ gibt den Status Ihrer Integration an.

Eine Integration ändern

Sie können eine bestehende Integration ändern.

1. Wählen Sie in der oberen rechten Ecke Ihrer Integrationsdetailseite Bearbeiten aus.
2. Auf der Seite Quelle und Ziel bearbeiten können Sie die Target-IAM-Rolle und die Target-Netzwerkverbindung ändern. Die anderen Felder können nach der Erstellung der Integration nicht bearbeitet werden. Klicken Sie auf Weiter.
3. Sie können den Namen und die Beschreibung der Integration auch auf der Seite Integration und Konfiguration bearbeiten bearbeiten. Klicken Sie auf Weiter.

4. Überprüfen Sie Ihre Änderungen und klicken Sie nach der Bestätigung auf Integration aktualisieren.

Eine Integration löschen

Löschen ist ein Endstatus für eine Integration. Nach dem Löschen kann die Integration nicht wiederbelebt werden. Durch das Löschen einer Integration werden alle internen Metadaten und alle zwischengespeicherten Daten gelöscht.

Während dieses Vorgangs werden alle laufenden Aufgaben, die das Schreiben von Daten in eine Zieltabelle beinhalten, beendet. AWS Glue löscht oder bereinigt die AWS Glue Zieldatenbank (im Datenkatalog) und die zugehörigen Daten im Amazon S3 S3-Bucket in Ihrem Konto nicht. Sie müssen diese bei Bedarf explizit bereinigen.

Um eine Integration zu löschen:

1. Klicken Sie auf der Seite mit den Integrationsdetails auf Löschen.
2. Geben Sie „Löschen“ ein und klicken Sie auf Löschen. Hinweis: Diese Aktion kann nicht rückgängig gemacht werden.
3. Auf der Seite mit den Integrationsdetails wird der Status „Löschen“ angezeigt. Sobald die Integration tatsächlich gelöscht wurde, wird sie nicht mehr auf der Homepage der Zero ETL-Integration angezeigt.

Erstellen von Integrationen mit dem APIs

Sie können Folgendes verwenden, APIs um Zero-ETL-Integrationen zu erstellen und zu verwalten in: AWS Glue

- CreateIntegration
- CreateIntegrationTableProperties
- CreateIntegrationResourceProperty
- UpdateIntegrationTableProperties
- UpdateIntegrationResourceProperty
- ModifyingIntegration
- DeleteIntegration

- DeleteIntegrationTableProperties
- DescribeIntegrations
- DescribeInboundIntegrations
- GetIntegrationTableProperties
- GetIntegrationResourceProperty

Weitere Informationen finden Sie unter Integration APIs in AWS Glue.

Überwachung einer Integration

Integrationsstaaten

Die folgenden Integrationszustände beschreiben die Integration:

- `Creating`- Die Integration wird erstellt.
- `Active`- Die Integration sendet Transaktionsdaten an das Ziel.
- `Modifying`- Die Integration wird geändert.
- `Syncing`- Bei der Integration ist ein behebbarer Fehler aufgetreten und die Daten werden erneut abgerufen.
- `Needs attention`- Bei der Integration ist ein Ereignis oder ein Fehler aufgetreten, für dessen Behebung ein manuelles Eingreifen erforderlich ist. Folgen Sie den Anweisungen in der Fehlermeldung in den Integrationsdetails, um das Problem zu beheben.
- `Failed`- Bei der Integration ist ein Ereignis oder ein Fehler aufgetreten, der nicht behebbar ist. Sie müssen die Integration löschen und erneut erstellen.
- `Deleting`- Die Integration wird gelöscht.

CloudWatch Amazon-Logs für eine Integration anzeigen

AWS Glue Zero-ETL-Integrationen generieren CloudWatch Amazon-Protokolle, um Einblick in Ihre Datenbewegungen zu erhalten. Protokollereignisse, die sich auf jede erfolgreiche Erfassung oder auf Fehler beziehen, die auf problematische Datensätze an der Quelle oder auf Fehler beim Schreiben von Daten aufgrund von Schemaänderungen oder unzureichenden Berechtigungen zurückzuführen sind, werden an eine Standard-Protokollgruppe gesendet, die in einem Kundenkonto erstellt wurde.

Für jede erstellte Integration werden die Protokollereignisse für diese Integration `/aws-glue/zeroETL-integrations/logs/` in Amazon Cloudwatch gesammelt. Innerhalb der Protokollgruppe werden Protokollnachrichten in Protokollstreams aufgeteilt. Jede erstellte Integration hat einen eigenen Protokollstream, in den alle Protokolle für diese Integration geschrieben werden. Logs für eine Integration mit finden Sie beispielsweise unter /aws- IntegrationArn `arn:aws:glue:us-east-1:123456789012:integration:03cabe77-79e7-4b7a-b3da-8c160bea6bbf` glue/zeroETL-integrations/logs /03cabe77-79e7-4b7a-b3da-8c160bea6bbf. {IntegrationId} kann aus dem {IntegrationArn} referenziert werden, der bei der Erstellung einer Integration generiert wird.

> **Note**
>
> In einem kontenübergreifenden Szenario werden Quellverarbeitungsprotokolle in dem Quellkonto ausgegeben, in dem die Integration vorhanden ist, und Zielverarbeitungsprotokolle werden in dem Zielkonto ausgegeben, in dem die Zieldatenbank vorhanden ist.

Für die Aktivierung der Protokollierung sind IAM-Berechtigungen erforderlich

Bei der Erstellung Ihrer Integration benötigen die Quell- und Zielrollen die folgenden IAM-Berechtigungen, um die CloudWatch Protokollierung für eine Integration zu aktivieren. AWS Glue Zero-ETL-Integrationen verwenden diese in den Quell- und Zielrollen bereitgestellten Berechtigungen, um CloudWatch Protokolle an Kundenkonten zu senden.

```
{
"Version": "2012-10-17",
    "Statement": [
        {
"Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
             ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Protokollmeldungen

Protokollformat: Zero-ETL-Integrationen geben vier Arten von Protokollnachrichten aus:

```
// Ingestion started
{
"integrationArn": "arn:aws:glue:us-east-2:123456789012:integration/1a012bba-123a-1bba-
ab1c-173de3b12345",
...
    "messageType": "IngestionStarted",
    "details": {
        "tableName": "testDDBTable",
        "message": "Ingestion Job started"
    }
}
// Data processing stats on successful table ingestion
{
...
    "messageType": "IngestionProcessingStats",
    "details": {
        "tableName": "testDDBTable",
        "insert_count": 100,
        "update_count": 10,
        "delete_count": 10
    }
}
// Ingestion failure logs for failed table-processing
{
...
    "messageType": "IngestionFailed",
    "details": {
        "tableName": "testDDBTable",
        "errorMessage": "Failed to ingest data with error: Target Glue database not
 found.",
        "error_code" : "client_error"
    }
}
// Ingestion completed notification with lastSyncedTimestamp
{
...
    "messageType": "IngestionCompleted",
    "details": {
        "tableName": "testDDBTable",
        "message": "Ingestion Job completed"
```

```
        "lastSyncedTimestamp": "1132344255745"
    }
}
```

CloudWatch Amazon-Metriken für eine Integration anzeigen

Sobald eine Integration abgeschlossen ist, können Sie die folgenden Amazon Cloudwatch-Metriken sehen, die in Ihrem Konto für jeden ausgeführten AWS Glue Job generiert wurden:

CloudWatch Namespace für Metriken: "AWS/glue/ZeroETL“

Dimensionen der Metriken:

- `integrationArn`
- `loadType`
- `tableName`

Namen der Metriken:

- `InsertCount`- Anzahl der Datensätze, die in die Iceberg-Zieltabelle eingefügt wurden.
- `UpdateCount`- Anzahl der in der Ziel-Iceberg-Tabelle aktualisierten Datensätze.
- `DeleteCount`- Anzahl der aus der Iceberg-Zieltabelle gelöschten Datensätze.
- `IngestionSucceeded`- zählt 1, wenn die Aufnahme für die Integration erfolgreich war.
- `IngestionFailed`- zählt 1, wenn die Aufnahme für die Integration fehlgeschlagen ist.
- `LastSyncTimestamp`- Zeitstempel, bis zu welcher Quelle mit dem Ziel synchronisiert wurde.

Verwaltung von Event-Benachrichtigungen mit Amazon EventBridge

Zero-ETL-Integrationen verwenden Amazon EventBridge zur Verwaltung von Ereignisbenachrichtigungen, damit Sie up-to-date über Änderungen an Ihren Integrationen informiert werden. Amazon EventBridge ist ein serverloser Event-Bus-Service, mit dem Sie Ihre Anwendungen mit Daten aus einer Vielzahl von Quellen verbinden können. In diesem Fall ist AWS Glue die Ereignisquelle. Ereignisse, bei denen es sich um EventBridge überwachte Änderungen in einer Umgebung handelt, werden AWS Glue automatisch gesendet. Ereignisse werden nahezu in Echtzeit übermittelt.

EventBridge bietet eine Umgebung, in der Sie Ereignisregeln schreiben können, die Aktionen festlegen können, die für bestimmte Ereignisse ergriffen werden sollen. Sie können auch Ziele einrichten, d. h. Ressourcen, an die ein Ereignis gesendet werden EventBridge kann. Ein Ziel kann ein API-Ziel, eine CloudWatch Amazon-Protokollgruppe und andere enthalten. Weitere Informationen zu Regeln finden Sie unter EventBridge Amazon-Regeln. Weitere Informationen zu Zielen finden Sie unter EventBridge Amazon-Ziele.

Um alle Zero-ETL-Benachrichtigungen zu erfassen, erstellen Sie eine Eventbridge-Regel, die den folgenden Kriterien entspricht:

```
{
  "source": [{
    "prefix": "aws.glue-zero-etl“
  }],
  "detail-type": [{
    "prefix": "Glue Zero ETL“
  }]
}
```

Die folgende Tabelle enthält Ereignisse der Null-ETL-Integration mit zusätzlichen Metadaten:

Detailtyp für Kunden	Erklärung
Glue Zero ETL-Aufnahme abgeschlossen	Die individuelle Ausführung für eine Entität wurde erfolgreich abgeschlossen.
Glue Zero ETL-Aufnahme ist fehlgeschlagen	Die individuelle Ausführung einer Entität wurde erfolglos abgeschlossen (entweder aufgrund eines Client- oder Systemfehlers).
Glue Zero ETL-Integration erneut synchronisiert	Die Integration wurde ERNEUT SYNCHRONISIERT.
Die ETL-Integration von Glue Zero ist fehlgeschlagen	Der Integrationsstatus wurde aufgrund eines Fehlers auf FEHLGESCHLAGEN geändert.
Die ETL-Integration von Glue Zero erfordert Aufmerksamkeit	Der Integrationsstatus wurde aufgrund eines Fehlers in NEEDS_ATTENTION geändert.

Einschränkungen

Im Folgenden finden Sie allgemeine Einschränkungen oder Überlegungen zu Zero-ETL-Integrationen:

- Ressourceneigenschaften stehen in einer one-to-one Beziehung zur entsprechenden Ressource. Folglich müssen alle Integrationen, die mit dieser Ressource erstellt wurden, der singulären Ressourceneigenschaft entsprechen. Die Änderung einer Ressourceneigenschaft wirkt sich daher auf alle Integrationen aus, die mit dieser Ressource verknüpft sind.
- Tabelleneigenschaften stehen in einer one-to-one Beziehung zu der entsprechenden Tabelle oder dem entsprechenden Objekt innerhalb einer Ressource. Daher müssen alle Integrationen, die dieselbe Tabelle zu oder von derselben Ressource verarbeiten, der singulären Tabelleneigenschaft entsprechen.
- Sie können eine Spalte an der Quelle nicht umbenennen. Wenn eine Spalte umbenannt wird, gibt es keine Garantie dafür, dass die Schemaerkennung korrekt durchgeführt wird, AWS Glue und die Auswirkungen auf die Integration sind nicht definiert.
- Für die Funktionsweise der Integration mit AWS Lake Formation verwalteten Tabellen gilt Folgendes: Standardmäßig verwenden Sie die IAM/AWS Glue -Richtlinie, um Ihre Tabelle und Datenbanken zu verwalten.
- Wenn Sie die Tabellenerstellung in dieser Datenbank verwalten AWS Lake Formation möchten, müssen Sie sicherstellen, dass die Rolle über ausreichende Lake Formation Formation-Berechtigungen verfügt, um die Tabelle und die Datenbank zu erstellen, zu ändern und zu löschen.
- Die Zero-ETL-Übersichtsseite enthält derzeit keine Metriken.

Im Folgenden sind quellenspezifische Einschränkungen von Zero-ETL-Integrationen aufgeführt:

- Zero-ETL-Integrationen mit einer SAP-Quelle unterstützen jetzt Entitäten, die mit beginnen. OData `Entity0f` Die Möglichkeit, den Primärschlüssel zu überschreiben, wird derzeit nur für Objekte unterstützt. SAPOData `Entity0f` Sobald diese Eigenschaft festgelegt wurde, kann sie nicht mehr geändert werden.
- Zero-ETL-Integrationen von Amazon DynamoDB zu Amazon SageMaker Lakehouse (über S3) unterstützen eine maximale DynamoDB-Tabellengröße von 50 TB.
- Die DynamoDB-Quelltabelle muss entweder mit einem Amazon-eigenen oder einem vom Kunden verwalteten KMS-Schlüssel verschlüsselt werden. AWS Die von Amazon verwaltete Verschlüsselung wird für die DynamoDB-Quelltabelle nicht unterstützt.

- SAP OData verwendet ein Delta-Token, bei dem die Kombination aus einem OAuth Client und einer Entität oder einer Standardauthentifizierung plus einer Entität nur ein einziges Delta-Token haben kann. Vermeiden Sie es, dieselbe Entität in zwei verschiedenen Integrationen mit demselben Client zu verwenden.
- Die folgenden Salesforce-Entitäten oder -Felder werden für die Verwendung in einer Zero-ETL-Integration mit einer Salesforce-Quelle nicht unterstützt.

```
AccountChangeEvent, AccountContactRoleChangeEvent, AccountHistory, AccountShare,
 ActiveFeatureLicenseMetric, ActivePermSetLicenseMetric, ActiveProfileMetric,
 ActivityFieldHistory, amzsec__asi_Telemetry_Data_Store__ChangeEvent,
 amzsec__asi_Telemetry_Data_Store__History, amzsec__asi_Telemetry_Data_Store__Share,
 amzsec__asi_Telemetry_Job_Log__ChangeEvent, amzsec__asi_Telemetry_Job_Log__History,
 amzsec__asi_Telemetry_Job_Log__Share,
 amzsec__asi_Telemetry_Requirement__ChangeEvent,
 amzsec__asi_Telemetry_Requirement__History,
 amzsec__asi_Telemetry_Requirement__Share, ApexClass, ApexComponent, ApexLog,
 ApexPage, ApexTestQueueItem, ApexTestResult, ApexTrigger, AssetChangeEvent,
 AssetHistory, AssetRelationshipHistory, AssetShare, AssignmentRule,
 AssociatedLocationHistory, AsyncApexJob, AuditTrailFileExportShare,
 AuthorizationFormConsentChangeEvent, AuthorizationFormConsentHistory,
 AuthorizationFormConsentShare, AuthorizationFormDataUseHistory,
 AuthorizationFormDataUseShare, AuthorizationFormHistory, AuthorizationFormShare,
 AuthorizationFormTextHistory, AuthProvider, AuthSession, BatchJobHistory,
 BatchJobPartFailedRecordHistory, BatchJobPartHistory, BatchJobShare,
 BrandTemplate, BriefcaseAssignmentChangeEvent, BriefcaseDefinitionChangeEvent,
 BusinessBrandShare, BusinessHours, BusinessProcess, CalcMatrixColumnRangeHistory,
 CalcProcStepRelationshipHistory, CalculationMatrixColumnHistory,
 CalculationMatrixHistory, CalculationMatrixRowHistory, CalculationMatrixShare,
 CalculationMatrixVersionHistory, CalculationProcedureHistory,
 CalculationProcedureShare, CalculationProcedureStepHistory,
 CalculationProcedureVariableHistory, CalculationProcedureVersionHistory, Calendar,
 CalendarViewShare, CallCenter, CallCoachConfigModifyEvent, CampaignChangeEvent,
 CampaignHistory, CampaignMemberChangeEvent, CampaignMemberStatusChangeEvent,
 CampaignShare, CaseChangeEvent, CaseHistory, CaseHistory2 CaseHistory2ChangeEvent,
 CaseRelatedIssueChangeEvent, CaseRelatedIssueHistory, CaseShare, CaseStatus,
 CaseTeamMember, CaseTeamRole, CaseTeamTemplate, CaseTeamTemplateMember,
 CaseTeamTemplateRecord, CategoryNode, ChangeRequestChangeEvent,
 ChangeRequestHistory, ChangeRequestRelatedIssueChangeEvent,
 ChangeRequestRelatedIssueHistory, ChangeRequestRelatedItemChangeEvent,
 ChangeRequestRelatedItemHistory, ChangeRequestShare, ChatRetirementRdyMetrics,
 ChatterActivity, ClientBrowser, CollaborationGroup, CollaborationGroupMember,
 CollaborationGroupMemberRequest, CollaborationInvitation,
```

```
CommSubscriptionChannelTypeHistory, CommSubscriptionChannelTypeShare,
CommSubscriptionConsentChangeEvent, CommSubscriptionConsentHistory,
CommSubscriptionConsentShare, CommSubscriptionHistory, CommSubscriptionShare,
CommSubscriptionTimingHistory, Community, ConnectedApplication, ContactChangeEvent,
ContactHistory, ContactPointAddressChangeEvent, ContactPointAddressHistory,
ContactPointAddressShare, ContactPointConsentChangeEvent,
ContactPointConsentHistory, ContactPointConsentShare,
ContactPointEmailChangeEvent, ContactPointEmailHistory, ContactPointEmailShare,
ContactPointPhoneChangeEvent, ContactPointPhoneHistory, ContactPointPhoneShare,
ContactPointTypeConsentChangeEvent, ContactPointTypeConsentHistory,
ContactPointTypeConsentShare, ContactRequestShare, ContactShare,
ContentDocumentChangeEvent, ContentDocumentHistory, ContentDocumentLink,
ContentDocumentLinkChangeEvent, ContentDocumentSubscription, ContentFolderItem,
ContentFolderLink, ContentFolderMember, ContentNote, ContentNotification,
ContentTagSubscription, ContentUserSubscription, ContentVersionChangeEvent,
ContentVersionComment, ContentVersionHistory, ContentVersionRating,
ContentWorkspace, ContentWorkspaceMember, ContentWorkspacePermission,
ContentWorkspaceSubscription, ContractChangeEvent, ContractHistory,
ContractLineItemChangeEvent, ContractLineItemHistory, ContractStatus,
Conversation, ConversationParticipant, CronJobDetail, CronTrigger, CustomBrand,
CustomBrandAsset, CustomerShare, CustomHttpHeader, DashboardComponent,
DataUseLegalBasisHistory, DataUseLegalBasisShare, DataUsePurposeHistory,
DataUsePurposeShare, DecisionTableRecordset, DeleteEvent, DocumentAttachmentMap,
Domain, DomainSite, DTRecordsetReplicaShare, EmailBounceEvent,
EmailMessageChangeEvent, EmailServicesAddress, EmailServicesFunction,
EmailTemplate, EmailTemplateChangeEvent, EngagementAttendeeChangeEvent,
EngagementAttendeeHistory, EngagementChannelTypeHistory, EngagementChannelTypeShare,
EngagementInteractionChangeEvent, EngagementInteractionHistory,
EngagementInteractionShare, EngagementInterface, EngagementTopicChangeEvent,
EngagementTopicHistory, EntitlementChangeEvent, EntitlementHistory,
EntitlementTemplate, EntityMilestoneHistory, EntitySubscription, EventChangeEvent,
EventRelationChangeEvent, EventRelayConfigChangeEvent, ExpressionSetHistory,
ExpressionSetShare, ExpressionSetVersionHistory, ExternalEventMappingShare,
FeedAttachment, FeedLike, FeedPollChoice, FeedPollVote, FeedSignal,
FieldPermissions, FieldSecurityClassification, FiscalYearSettings,
FlowInterviewLogShare, FlowInterviewShare, FlowOrchestrationEvent,
FlowOrchestrationInstanceShare, FlowOrchestrationStageInstanceShare,
FlowOrchestrationStepInstanceShare, FlowOrchestrationWorkItemShare,
FlowRecordShare, FlowRecordVersionChangeEvent, FlowTestResultShare, Folder,
Group, GroupMember, Holiday, IdeaComment, IdpEventLog, ImageHistory, ImageShare,
IncidentChangeEvent, IncidentHistory, IncidentRelatedItemChangeEvent,
IncidentRelatedItemHistory, IncidentShare, IndividualChangeEvent,
IndividualHistory, IndividualShare, KnowledgeableUser, LeadChangeEvent, LeadHistory,
LeadShare, LeadStatus, LightningExitByPageMetrics, LightningToggleMetrics,
```

```
LightningUsageByAppTypeMetrics, LightningUsageByBrowserMetrics,
LightningUsageByFlexiPageMetrics, LightningUsageByPageMetrics, ListEmailChangeEvent,
ListEmailShare, ListView, LocationChangeEvent, LocationHistory, LocationShare,
LocationTrustMeasureShare, LoginHistory, LoginIp, MacroChangeEvent,
MacroHistory, MacroInstructionChangeEvent, MacroShare, MacroUsageShare,
ManagedContentVariantChangeEvent, MessagingEndUserHistory, MessagingEndUserShare,
MessagingSessionHistory, MessagingSessionShare, MilestoneType, MLEngagementEvent,
ObjectPermissions, OpportunityChangeEvent, OpportunityContactRoleChangeEvent,
OpportunityFieldHistory, OpportunityLineItemChangeEvent, OpportunityShare,
OpportunityStage, OrderChangeEvent, OrderHistory, OrderItemChangeEvent,
OrderItemHistory, OrderShare, OrderStatus, Organization, OrgEmailAddressSecurity,
OrgWideEmailAddress, OutgoingEmail, OutgoingEmailRelation,
PackageLicense, PartnerRole, PartyConsentChangeEvent, PartyConsentHistory,
PartyConsentShare, Period, PermissionSet, PermissionSetAssignment,
PermissionSetTabSetting, Pricebook2ChangeEvent, Pricebook2History,
PricebookEntryChangeEvent, PricebookEntryHistory, PrivacyJobSessionShare,
PrivacyObjectSessionShare, PrivacyRTBFRequestHistory, PrivacyRTBFRequestShare,
PrivacySessionRecordFailureShare, ProblemChangeEvent, ProblemHistory,
ProblemIncidentChangeEvent, ProblemIncidentHistory, ProblemRelatedItemChangeEvent,
ProblemRelatedItemHistory, ProblemShare, ProcessDefinition, ProcessExceptionEvent,
ProcessExceptionShare, ProcessInstanceChangeEvent, ProcessInstanceStep,
ProcessInstanceStepChangeEvent, ProcessNode, Product2ChangeEvent, Product2History,
ProductEntitlementTemplate, Profile, ProfileSkillEndorsementHistory,
ProfileSkillHistory, ProfileSkillShare, ProfileSkillUserHistory, PromptActionShare,
PromptErrorShare, QueueSobject, QuickTextChangeEvent, QuickTextHistory,
QuickTextShare, QuickTextUsageShare, RecentlyViewed, RecommendationChangeEvent,
RecordActionHistory, RecordAlertHistory, RecordAlertShare, RecordType,
ScorecardShare, SellerHistory, SellerShare, ServiceContractChangeEvent,
ServiceContractHistory, ServiceContractShare, SetupAuditTrail, SetupEntityAccess,
SharingRecordCollectionShare, Site, SiteHistory, SiteRedirectMapping,
SocialPersonaHistory, SocialPostChangeEvent, SocialPostHistory, SocialPostShare,
SolutionHistory, SolutionStatus, StaticResource, StreamingChannelShare,
TableauHostMappingShare, TaskChangeEvent, TaskPriority, TaskStatus,
ThreatDetectionFeedback, TimelineObjectDefinitionChangeEvent, TodayGoalShare,
Topic, TopicUserEvent, Translation, User, UserAppMenuCustomizationShare,
UserChangeEvent, UserDefinedLabelAssignmentShare, UserDefinedLabelShare,
UserEmailPreferredPersonShare, UserLicense, UserLogin, UserPackageLicense,
UserPreference, UserPrioritizedRecordShare, UserProvisioningRequestShare,
UserRole, UserShare, VideoCallChangeEvent, VideoCallParticipantChangeEvent,
VideoCallRecordingChangeEvent, VideoCallShare, VisualforceAccessMetrics,
VoiceCallChangeEvent, VoiceCallRecordingChangeEvent, VoiceCallShare, Vote,
WebLink, WorkAccessShare, WorkBadgeDefinitionHistory, WorkBadgeDefinitionShare,
WorkOrderChangeEvent, WorkOrderHistory, WorkOrderLineItemChangeEvent,
WorkOrderLineItemHistory, WorkOrderLineItemStatus, WorkOrderShare,
```

```
 WorkOrderStatus, WorkPlanChangeEvent, WorkPlanHistory, WorkPlanShare,
 WorkPlanTemplateChangeEvent, WorkPlanTemplateEntryChangeEvent,
 WorkPlanTemplateEntryHistory, WorkPlanTemplateHistory, WorkPlanTemplateShare,
 WorkStepChangeEvent, WorkStepHistory, WorkStepStatus, WorkStepTemplateChangeEvent,
 WorkStepTemplateHistory, WorkStepTemplateShare, WorkThanksShare
```

- Die folgenden ServiceNow Entitäten oder Felder werden für die Verwendung in einer Zero-ETL-Integration mit einer Quelle nicht unterstützt. ServiceNow

```
ais_acl_overrides, ais_async_genius_result, ais_async_request,
 ais_connection, ais_genius_result_configuration_parameters,
 ais_partition_health, ais_partition_health_response, ais_publish_history,
 ais_relevancy_training_execution, ais_relevancy_training_staging,
 ais_search_profile_relevancy_model, catalog_draft_entities, clone_log,
 clone_log0000, clone_log0001, clone_log0002, clone_log0003, clone_log0004,
 clone_log0005, clone_log0006, clone_log0007, cmdb_ie_context, cmdb_ie_log,
 cmdb_ie_run, cmdb_ire_partial_payloads_index, cmdb_qb_result_base
$par1, cmdb$par1, discovery_log, discovery_log0000, discovery_log0001,
 discovery_log0002, discovery_log0003, discovery_log0004, discovery_log0005,
 discovery_log0006, discovery_log0007, ecc_agent_log, entitlement_data,
 entl_subscription_map, gs_entitlement_plugin_mapping, ih_transaction_exclusion,
 import_log, import_log0000, import_log0001, import_log0002, import_log0003,
 import_log0004, import_log0005, import_log0006, import_log0007, jrobin_archive,
 jrobin_database, jrobin_datasource, jrobin_definition, jrobin_graph,
 jrobin_graph_line, jrobin_graph_set, jrobin_graph_set_member, jrobin_shard,
 jrobin_shard_location, license_role_discovery_run, logger_configuration_validation,
 m2m_analytics_event_logger, m2m_user_consent_info, ml_artifact_object_store,
 np$sys_gen_ai_filter_sample, np$sys_ui_element, np$sys_ui_list_element,
 one_api_service_plan_feature_invocation, one_api_service_plan_invocation,
 open_nlu_predict_log, open_nlu_predict_log0000, open_nlu_predict_log0001,
 open_nlu_predict_log0002, open_nlu_predict_log0003, open_nlu_predict_log0004,
 open_nlu_predict_log0005, open_nlu_predict_log0006, open_nlu_predict_log0007,
 pa_diagnostic_log, pa_diagnostic_log0000, pa_diagnostic_log0001,
 pa_diagnostic_log0002, pa_diagnostic_log0003, pa_diagnostic_log0004,
 pa_diagnostic_log0005, pa_diagnostic_log0006, pa_diagnostic_log0007, pa_favorites,
 pa_job_log_rows, pa_job_log_rows0000, pa_job_log_rows0001, pa_job_log_rows0002,
 pa_job_log_rows0003, pa_job_log_rows0004, pa_job_log_rows0005, pa_job_log_rows0006,
 pa_job_log_rows0007, pa_migration_ignored_scores, pa_scores_l1, pa_scores_l2,
 pa_scores_migration_groups, par_dashboard_conversion_backup, promin_log,
 promin_log0000, promin_log0001, promin_log0002, promin_log0003, promin_log0004,
 promin_log0005, promin_log0006, promin_log0007, promin_request_object,
 proposed_change_verification_log, proposed_change_verification_log0000,
 proposed_change_verification_log0001, proposed_change_verification_log0002,
```

```
proposed_change_verification_log0003, proposed_change_verification_log0004,
proposed_change_verification_log0005, proposed_change_verification_log0006,
proposed_change_verification_log0007, protected_table_log, protected_table_log0000,
protected_table_log0001, protected_table_log0002, protected_table_log0003,
protected_table_log0004, protected_table_log0005, protected_table_log0006,
protected_table_log0007, pwd_history, qb_query_results, scan_log,
scan_log0000, scan_log0001, scan_log0002, scan_log0003, scan_log0004,
scan_log0005, scan_log0006, scan_log0007, schema_validator_error,
sla_repair_log_entry, sla_repair_log_entry0000, sla_repair_log_entry0001,
sla_repair_log_entry0002, sla_repair_log_entry0003, sla_repair_log_entry0004,
sla_repair_log_entry0005, sla_repair_log_entry0006, sla_repair_log_entry0007,
sla_repair_log_message, sla_repair_log_message0000, sla_repair_log_message0001,
sla_repair_log_message0002, sla_repair_log_message0003, sla_repair_log_message0004,
sla_repair_log_message0005, sla_repair_log_message0006, sla_repair_log_message0007,
sn_bm_client_activity, sn_ci_analytics_st_actionable_notifs,
sn_ci_analytics_st_conv_completion_by_cat, sn_ci_analytics_st_conv_dynamic_property,
sn_ci_analytics_st_conversation, sn_ci_analytics_st_count_by_date,
sn_ci_analytics_st_event_occurrence, sn_ci_analytics_st_event_property_value_trend,
sn_ci_analytics_st_issue_auto_resolution, sn_ci_analytics_st_no_clicks,
sn_ci_analytics_st_no_results, sn_ci_analytics_st_property_summary_by_event,
sn_ci_analytics_st_session_count_per_locale, sn_ci_analytics_st_session_duration,
sn_ci_analytics_st_spokes_usage, sn_ci_analytics_st_topic_execution_stats,
sn_ci_analytics_st_topic_occurrence, sn_ci_analytics_st_trending_content,
sn_ci_analytics_st_trending_queries, sn_ci_analytics_st_users,
sn_cs_plugin_signatures, sn_cs_telemetry_log, sn_dfc_application, sn_dfc_product,
sn_employee_position, sn_entitlement_st_subscription_application_family,
sn_entitlement_st_subscription_application_users, sn_hr_sp_st_relevant_for_you,
sn_instance_clone_log, sn_instance_clone_log0000, sn_instance_clone_log0001,
sn_instance_clone_log0002, sn_instance_clone_log0003, sn_instance_clone_log0004,
sn_instance_clone_log0005, sn_instance_clone_log0006, sn_instance_clone_log0007,
sn_km_mr_st_kb_knowledge, sn_me_st_topic, sn_rf_conditional_definition,
sn_rf_evaluation_type, sn_rf_evaluation_type_input, sn_rf_recommendation_action,
sn_rf_recommendation_experience, sn_rf_recommendation_history,
sn_rf_recommendation_rule, sn_rf_record_display_configuration,
sn_rf_trend_definition, sn_sub_man_st_account_level_entitlement,
sn_sub_man_st_gen_ai_metadata, sn_sub_man_st_instance_used_assist_count,
sn_sub_man_st_now_assist_creator_instances, sn_sub_man_st_now_assists_aggregate,
sn_sub_man_st_subscribed_groups, sn_sub_man_st_subscription_insights,
sn_sub_man_st_subscription_license_detail_metric,
sn_sub_man_st_unallocated_group_recommendation,
sn_sub_man_st_unconfirmed_user_group, sn_wn_user_app_activity,
sn_wn_user_content_activity, snc_monitorable_item,
snpar_sched_export_v_scheduled_export_visualization, spotlight,
spotlight_audit, spotlight_copy_log_row, spotlight_copy_log_row0000,
```

```
spotlight_copy_log_row0001, spotlight_copy_log_row0002, spotlight_copy_log_row0003,
spotlight_copy_log_row0004, spotlight_copy_log_row0005, spotlight_copy_log_row0006,
spotlight_copy_log_row0007, spotlight_job_log_row, spotlight_job_log_row0000,
spotlight_job_log_row0001, spotlight_job_log_row0002, spotlight_job_log_row0003,
spotlight_job_log_row0004, spotlight_job_log_row0005, spotlight_job_log_row0006,
spotlight_job_log_row0007, st_dfc_performance_metric, st_license_detail_metric,
st_on_call_hour, st_sc_wizard_question, st_sys_catalog_items_and_variable_sets,
st_sys_design_system_icon, subscription_instance_stats,
svc_container_config, svc_environment_config, svc_layer_config,
svc_model_assoc_ci, svc_model_checkpoint_attr, svc_model_obj_cluster,
svc_model_obj_constraint, svc_model_obj_deployable, svc_model_obj_element,
svc_model_obj_impact, svc_model_obj_impactrule, svc_model_obj_package,
svc_model_obj_path, svc_model_obj_relation, svc_model_obj_service,
sys_administrative_script_transaction, sys_amb_message, sys_amb_processor,
sys_analytics_batch_state, sys_analytics_config, sys_analytics_data_points_error,
sys_analytics_event, sys_analytics_logger, sys_analytics_logger_field,
sys_app_payload_loader_rule, sys_app_payload_unloader_rule, sys_app_scan_payload,
sys_app_scan_variable, sys_app_scan_variable_type, sys_archive_destroy_log,
sys_archive_destroy_run, sys_archive_log, sys_archive_run, sys_atf_transaction_log,
sys_attachment_doc, sys_attachment_doc_v2, sys_attachment_soft_deleted, sys_audit,
sys_audit_relation, sys_auth_policy_api_allowed, sys_aw_registered_scripting_modal,
sys_cache_flush, sys_data_egress_source, sys_dm_delete_count,
sys_export_set_log, sys_export_set_log0000, sys_export_set_log0001,
sys_export_set_log0002, sys_export_set_log0003, sys_export_set_log0004,
sys_export_set_log0005, sys_export_set_log0006, sys_export_set_log0007,
sys_flow_compiled_flow, sys_flow_compiled_flow_chunk, sys_flow_context_chunk,
sys_flow_context_chunk_archive, sys_flow_context_inputs_chunk,
sys_flow_execution_history, sys_flow_log, sys_flow_log0000, sys_flow_log0001,
sys_flow_log0002, sys_flow_log0003, sys_flow_plan_context_binding,
sys_flow_report_doc, sys_flow_report_doc_chunk, sys_flow_report_doc_chunk_archive,
sys_flow_runtime_state_chunk, sys_flow_runtime_value_chunk,
sys_flow_subflow_plan_chunk, sys_flow_trigger_plan_chunk, sys_flow_val_listener,
sys_flow_value, sys_flow_value_chunk, sys_gen_ai_config_example,
sys_gen_ai_feature_mapping, sys_gen_ai_strategy_mapping, sys_gen_ai_usage_log,
sys_generative_ai_capability_definition, sys_generative_ai_log,
sys_generative_ai_response_validator, sys_generative_ai_validator,
sys_geo_routing, sys_geo_routing_config, sys_hop_token, sys_hub_action_plan_chunk,
sys_hub_snapshot_chunk, sys_journal_field, sys_journal_field_edit, sys_json_chunk,
sys_kaa_policy, sys_kaa_subidentity_assertion, sys_kaa_user_policy_mapping,
sys_mapplication, sys_mass_encryption_job, sys_notification_execution_log,
sys_notification_execution_log0000, sys_notification_execution_log0001,
sys_notification_execution_log0002, sys_notification_execution_log0003,
sys_notification_execution_log0004, sys_notification_execution_log0005,
sys_notification_execution_log0006, sys_notification_execution_log0007,
```

```
sys_nowmq_message, sys_nowmq_provider_param_definition, sys_orchestrator_action,
sys_pd_asset_configuration, sys_pd_context_chunk, sys_pd_context_log,
sys_pd_snapshot_chunk, sys_pd_trigger_license, sys_processing_framework_job,
sys_query_index_hint, sys_query_rewrite, sys_query_string_log,
sys_replication_queue, sys_replication_queue0, sys_replication_queue1,
sys_replication_queue2, sys_replication_queue3, sys_replication_queue4,
sys_replication_queue5, sys_replication_queue6, sys_replication_queue7,
sys_request_performance, sys_rollback_blacklisted, sys_rollback_conflict,
sys_rollback_incremental, sys_rollback_log, sys_rollback_log0000,
sys_rollback_log0001, sys_rollback_log0002, sys_rollback_log0003,
sys_rollback_log0004, sys_rollback_log0005, sys_rollback_log0006,
sys_rollback_log0007, sys_rollback_run, sys_rollback_schema_change,
sys_rollback_schema_conflict, sys_rollback_sequence, sys_scheduler_assignment,
sys_scheduler_memory_pressure_job_log, sys_script_adapter_rule,
sys_script_batch_adapter_rule, sys_search_source_filter, sys_service_authentication,
sys_signing_job, sys_suggestion_reader, sys_sync_history_review, sys_trend,
sys_unreferenced_preview, sys_unreferenced_record_rule, sys_upgrade_manifest,
sys_upgrade_state, sys_ux_asset_cache_buster, sys_ux_lib_component_prop,
sys_ux_lib_presource, sys_ux_page_action, sys_ux_page_action_binding,
sysevent_queue_runtime, syslog, syslog_app_scope0000, syslog_app_scope0001,
syslog_app_scope0002, syslog_app_scope0003, syslog_app_scope0004,
syslog_app_scope0005, syslog_app_scope0006, syslog_app_scope0007, syslog_email0000,
syslog_email0001, syslog_email0002, syslog_email0003, syslog_email0004,
syslog_email0005, syslog_email0006, syslog_email0007, syslog_transaction,
syslog_transaction0000, syslog_transaction0001, syslog_transaction0002,
syslog_transaction0003, syslog_transaction0004, syslog_transaction0005,
syslog_transaction0006, syslog_transaction0007, syslog0000, syslog0001,
syslog0002, syslog0003, syslog0004, syslog0005, syslog0006, syslog0007,
ts_attachment, ts_c_1_0, ts_c_1_1, ts_c_1_2, ts_c_1_3, ts_c_1_4, ts_c_1_5,
ts_c_1_6, ts_c_1_7, ts_c_1_8, ts_c_1_9, ts_c_10_0, ts_c_10_1, ts_c_10_2,
ts_c_10_3, ts_c_10_4, ts_c_10_5, ts_c_10_6, ts_c_10_7, ts_c_10_8, ts_c_10_9,
ts_c_11_0, ts_c_11_1, ts_c_11_2, ts_c_11_3, ts_c_11_4, ts_c_11_5, ts_c_11_6,
ts_c_11_7, ts_c_11_8, ts_c_11_9, ts_c_2_0, ts_c_2_1, ts_c_2_2, ts_c_2_3,
ts_c_2_4, ts_c_2_5, ts_c_2_6, ts_c_2_7, ts_c_2_8, ts_c_2_9, ts_c_3_0,
ts_c_3_1, ts_c_3_2, ts_c_3_3, ts_c_3_4, ts_c_3_5, ts_c_3_6, ts_c_3_7, ts_c_3_8,
ts_c_3_9, ts_c_39_0, ts_c_39_1, ts_c_39_2, ts_c_39_3, ts_c_39_4, ts_c_39_5,
ts_c_39_6, ts_c_39_7, ts_c_39_8, ts_c_39_9, ts_c_40_0, ts_c_40_1, ts_c_40_2,
ts_c_40_3, ts_c_40_4, ts_c_40_5, ts_c_40_6, ts_c_40_7, ts_c_40_8, ts_c_40_9,
ts_c_41_0, ts_c_41_1, ts_c_41_2, ts_c_41_3, ts_c_41_4, ts_c_41_5, ts_c_41_6,
ts_c_41_7, ts_c_41_8, ts_c_41_9, ts_c_42_0, ts_c_42_1, ts_c_42_2, ts_c_42_3,
ts_c_42_4, ts_c_42_5, ts_c_42_6, ts_c_42_7, ts_c_42_8, ts_c_42_9, ts_c_43_0,
ts_c_43_1, ts_c_43_2, ts_c_43_3, ts_c_43_4, ts_c_43_5, ts_c_43_6, ts_c_43_7,
ts_c_43_8, ts_c_43_9, ts_c_44_0, ts_c_44_1, ts_c_44_2, ts_c_44_3, ts_c_44_4,
ts_c_44_5, ts_c_44_6, ts_c_44_7, ts_c_44_8, ts_c_44_9, ts_c_45_0, ts_c_45_1,
```

```
ts_c_45_2, ts_c_45_3, ts_c_45_4, ts_c_45_5, ts_c_45_6, ts_c_45_7, ts_c_45_8,
ts_c_45_9, ts_c_6_0, ts_c_6_1, ts_c_6_2, ts_c_6_3, ts_c_6_4, ts_c_6_5,
ts_c_6_6, ts_c_6_7, ts_c_6_8, ts_c_6_9, ts_c_7_0, ts_c_7_1, ts_c_7_2, ts_c_7_3,
ts_c_7_4, ts_c_7_5, ts_c_7_6, ts_c_7_7, ts_c_7_8, ts_c_7_9, ts_c_8_0, ts_c_8_1,
ts_c_8_2, ts_c_8_3, ts_c_8_4, ts_c_8_5, ts_c_8_6, ts_c_8_7, ts_c_8_8, ts_c_8_9,
ts_c_attachment, ts_chain, ts_deleted_doc, ts_document, ts_field, ts_index_stats,
ts_phrase, ts_search_stats, ts_v4_attachment, ts_word, ua_app_metadata,
ua_audit_stats, ua_extra_page, ua_monitor_property, ua_monitor_property_audit,
ua_shared_service, ua_sn_table_inventory, ua_sp_known_bot, ua_upload_log,
v_ais_result_improvement_rule_condition_builder_values, v_cluster_nodes,
v_cxs_search_resource, v_db_index, v_db_trigger, v_file_load_order,
v_interaction_context, v_iostats, v_mysql_proclist, v_mysql_status,
v_mysql_variables, v_on_call_report_cache, v_pa_par_combined_dashboard,
v_pd_activity_condition_to_run, v_pd_activity_start_rule_with_condition,
v_pd_lane_condition_to_run, v_sql_debug, v_st_kb_category, v_st_kb_most_viewed,
v_st_kb_recently_viewed, v_st_km_genai_mra_similar_task, v_st_popular_item,
v_st_recent_item, v_st_sc_cat_item, v_st_sc_catalog, v_st_sc_category,
validator_run_summary, vtb_card_history, wf_log, wf_log0000, wf_log0001,
wf_log0002, wf_log0003, wf_log0004, wf_log0005, wf_log0006, wf_log0007,
spotlight_criteria, pa_snapshots, pa_widget_indicators, pa_widgets,
sn_cim_register, global, gsw_change_log, gsw_content, gsw_content_group,
gsw_content_information, gsw_status_of_content, multi_factor_browser_fingerprint,
multi_factor_criteria, pa_filters, password_policy, plan_execution,
plan_mysql, plan_oracle, plan_postgres, sc_rest_api_without_access_policy,
sn_actsub_activity, sn_actsub_activity_fanout, sn_actsub_activity_stream,
sn_actsub_activity_type, sn_actsub_atype_attributes, sn_actsub_atype_notif_pref,
sn_actsub_module, sn_actsub_notif_object, sn_actsub_subobject_stream,
sn_actsub_subscribable_object, sn_actsub_subscription_notif_pref,
sn_actsub_user_stream, sn_appclient_store_outbound_http_quota,
sn_appcreator_app_template, sn_critical_update, sn_docker_spoke_images,
sn_employee_app, sn_employee_app_access, sn_employee_app_access_criteria,
sn_entitlement_genai_assist_counts, sn_entitlement_genai_creator_user_counts,
sn_entitlement_genai_creator_users, sn_mif_instance, sn_mif_sync_data,
sn_mif_sync_status, sn_mif_table_registration, sn_mif_trust_config,
sn_vsc_best_practice_configurations, sn_vsc_best_practice_goals,
sn_vsc_changed_hardening_settings, sn_vsc_changed_scan_findings,
sn_vsc_check_security_area, sn_vsc_elevation_event, sn_vsc_event,
sn_vsc_export_event, sn_vsc_export_setting, sn_vsc_harc_compliance_status_lookup,
sn_vsc_hardening_compliance_scores, sn_vsc_impersonation_event,
sn_vsc_instance_hardening_settings, sn_vsc_login_event,
sn_vsc_scan_comparisons, sn_vsc_scan_summary, sn_vsc_security_check_categories,
sn_vsc_security_check_configurations, sn_vsc_security_configuration_groups,
sn_vsc_security_privacy_capabilities, sn_vsc_updated_settings,
sn_vsc_user_comparisons, sys_app_hash_inventory, sys_coalesce_strategy_deferred,
```

```
sys_flow_secure_data, sys_formula_function, sys_geocoding_request,
sys_global_file_hash, sys_import_set_row, sys_index, sys_index_explain,
sys_installation_schedule, sys_installation_schedule_item, sys_offline_app,
sys_package, sys_package_dependency_item, sys_package_dependency_m2m,
sys_plugins, sys_querystat, sys_reap_package, sys_scoped_plugin,
sys_stage_storage_alias, sys_storage_alias, sys_storage_table_alias, sys_store_app,
sys_table_partition, sys_upgrade_history_log, sys_user_public_credential,
sys_webauthn_authentication_request, sys_webauthn_registration_request,
syslog_app_scope, syslog_email, syslog_page_timing, ua_instance_state_config,
v_expression_cache, v_private_cache, v_shared_cache, sys_amb_message0002,
sys_amb_message0004, sys_amb_message0005, sys_metadata, v_par_unified_report_viz
```

AWS Glue Qualität der Daten

AWS Glue Mit Data Quality können Sie die Qualität Ihrer Daten messen und überwachen, sodass Sie gute Geschäftsentscheidungen treffen können. AWS Glue Data Quality basiert auf dem DeeQu Open-Source-Framework und bietet ein verwaltetes, serverloses Erlebnis. AWS Glue Data Quality arbeitet mit der Data Quality Definition Language (DQDL), einer domänenspezifischen Sprache, mit der Sie Datenqualitätsregeln definieren. Weitere Informationen über DQDL und die unterstützten Regeltypen finden Sie unter Referenz zu Data Quality Definition Language (DQDL).

Weitere Produktdetails und Preise finden Sie auf der Serviceseite für AWS Glue Data Quality.

Vorteile und wichtige Features

Zu den Vorteilen und Hauptmerkmalen von AWS Glue Data Quality gehören:

- Serverlos — Es erfolgt keine Installation, kein Patching oder keine Wartung.
- Schneller Einstieg — AWS Glue Data Quality analysiert Ihre Daten schnell und erstellt Datenqualitätsregeln für Sie. Sie können mit zwei Klicks loslegen: „Datenqualitätsregeln erstellen → Regeln empfehlen“.
- Erkennen von Datenqualitätsproblemen — Verwenden Sie maschinelles Lernen (ML), um Anomalien und hard-to-detect Datenqualitätsprobleme zu erkennen.
- Improvisieren Sie Ihre Regeln — mit mehr als 25 out-of-the-box DQ-Regeln können Sie Regeln erstellen, die Ihren spezifischen Bedürfnissen entsprechen.
- Qualität bewerten und fundierte Geschäftsentscheidungen treffen – Sobald Sie die Regeln bewertet haben, erhalten Sie einen Datenqualitätswert, der einen Überblick über den Zustand Ihrer Daten bietet. Verwenden Sie den Wert von Data Quality, um sichere Geschäftsentscheidungen zu treffen.
- Schlechte Daten im Visier — AWS Glue Data Quality hilft Ihnen dabei, genau die Datensätze zu identifizieren, die zu einem Rückgang Ihrer Qualitätswerte geführt haben. Identifizieren, isolieren und beheben Sie sie ganz einfach.
- Pay-as-you-go — Für die Nutzung von AWS Glue Data Quality benötigen Sie keine Jahreslizenzen.
- Keine Bindung — AWS Glue Data Quality basiert auf Open Source DeeQu, sodass Sie die Regeln, die Sie erstellen, in einer offenen Sprache aufbewahren können.

- Datenqualitätsprüfungen — Sie können Datenqualitätsprüfungen an Data Catalog AWS Glue ETL-Pipelines durchsetzen, sodass Sie die Datenqualität im Ruhezustand und bei der Übertragung verwalten können.
- ML-basierte Datenqualitätserkennung — Verwenden Sie maschinelles Lernen (ML), um Anomalien und hard-to-detect Datenqualitätsprobleme zu erkennen.
- Offene Sprache für ausdrückliche Regeln — stellt sicher, dass Datenqualitätsregeln einheitlich und einfach verfasst werden. Geschäftsanwender können Datenqualitätsregeln einfach in einer einfachen Sprache ausdrücken, die sie verstehen können. Für Ingenieure bietet diese Sprache die Flexibilität, Code zu generieren, eine konsistente Versionskontrolle zu implementieren und Bereitstellungen zu automatisieren.

Funktionsweise

Es gibt zwei Einstiegspunkte für AWS Glue Datenqualität: die Jobs AWS Glue Data Catalog und AWS Glue ETL-Jobs. Dieser Abschnitt bietet einen Überblick über die Anwendungsfälle und AWS Glue Funktionen, die von den einzelnen Einstiegspunkten unterstützt werden.

Datenqualität für AWS Glue Data Catalog

AWS Glue Data Quality bewertet Objekte, die in der gespeichert sind. AWS Glue Data Catalog Es bietet Nicht-Programmierern eine einfache Möglichkeit, Datenqualitätsregeln einzurichten. Zu diesen Persönlichkeiten gehören Datenverwalter und Geschäftsanalysten.

Sie können diese Option für die folgenden Anwendungsfälle wählen:

- Sie möchten Datenqualitätsaufgaben für Datensätze durchführen, die Sie bereits im AWS Glue Data Catalog katalogisiert haben.
- Sie arbeiten an der Datenverwaltung und müssen kontinuierlich Datenqualitätsprobleme in Ihrem Data Lake identifizieren oder bewerten.

Sie können die Datenqualität für den Datenkatalog über die folgenden Schnittstellen verwalten:

- Die Management-Konsole AWS Glue
- AWS Glue APIs

Informationen zu den ersten Schritten mit AWS Glue Data Quality for the AWS Glue Data Catalog finden Sie unterErste Schritte mit AWS Glue Data Quality für die Data Catalog.

Datenqualität für AWS Glue ETL-Jobs

AWS Glue Mit Data Quality for AWS Glue ETL-Jobs können Sie proaktive Datenqualitätsaufgaben ausführen. Proaktive Aufgaben helfen Ihnen, fehlerhafte Daten zu identifizieren und herauszufiltern, bevor Sie einen Datensatz in Ihren Data Lake laden.

Video: Einführung in die AWS Glue Datenqualität für ETL-Pipelines

Sie können die Datenqualität für ETL-Aufträge für die folgenden Anwendungsfälle auswählen:

- Sie möchten Datenqualitätsaufgaben in Ihre ETL-Aufträge einbeziehen
- Sie möchten Code schreiben, der Datenqualitätsaufgaben in ETL-Skripten definiert
- Sie möchten die Qualität der Daten, die in Ihren visuellen Daten-Pipelines fließen, verwalten

Sie können die Datenqualität für ETL-Aufträge über die folgenden Schnittstellen verwalten:

- AWS Glue Studio, AWS Glue Studio Notizbücher und AWS Glue interaktive Sitzungen
- AWS Glue Bibliotheken für ETL-Scripting
- AWS Glue APIs

Informationen zu den ersten Schritten mit der Datenqualität für ETL-Aufträge finden Sie unter Tutorial: Erste Schritte mit Data Quality im AWS Glue Studio -Benutzerhandbuch.

Vergleich der Datenqualität für den Datenkatalog mit der Datenqualität für ETL-Aufträge

Diese Tabelle bietet einen Überblick über die Funktionen, die von den einzelnen Einstiegspunkten für AWS Glue Datenqualität unterstützt werden.

Funktion	Datenqualität für den Datenkatalog	Datenqualität für ETL-Aufträge
Datenquellen	Amazon S3, Amazon Redshift, mit dem Datenkatalog	Alle Datenquellen werden von unterstützt AWS Glue,

Funktion	Datenqualität für den Datenkatalog	Datenqualität für ETL-Aufträge
	kompatible JDBC-Quellen und transaktionale Data-Lake -Formate wie Apache Iceberg, Apache Hudi und Delta Lake. Beachten Sie, dass Iceberg-, Delta- und HUDI-Tabellen nicht unterstützt werden, wenn Tabellen AWS Lake Formation verwaltet werden. Amazon Athena katalogis ierte Ansichten werden nicht unterstützt AWS Glue Data Catalog .	einschließlich benutzerd efinierter Konnektoren und Konnektoren von Drittanbi etern.
Empfehlungen für Data-Qual ity-Regeln	Unterstützt	Nicht unterstützt
DQDL-Regeln erstellen und ausführen	Unterstützt	Unterstützt
Auto-Scaling	Nicht unterstützt	Unterstützt
AWS Glue Flex-Unterstützung	Nicht unterstützt	Unterstützt
Planung	Wird beim Auswerten von Data-Quality-Regeln und über Schrittfunktionen unterstützt.	Wird bei der Verwendung von Schrittfunktionen und Workflows unterstützt.
Identifizieren von Datensätz en, bei denen die Datenqual itätsprüfungen fehlgeschlagen sind	Nicht unterstützt	Unterstützt
Integration mit Amazon Eventbridge	Unterstützt	Unterstützt

Funktion	Datenqualität für den Datenkatalog	Datenqualität für ETL-Aufträge
Integration mit AWS Cloudwatch	Unterstützt	Unterstützt
Schreiben von Datenqualitätsergebnissen in Amazon S3	Unterstützt	Unterstützt
Inkrementelle Datenqualität	Wird über Pushdown-Prädikate unterstützt	Wird über AWS Glue Lesezeichen unterstützt
AWS CloudFormation Unterstützung	Unterstützt	Unterstützt
ML-gestützte Anomaliekennung	Nicht unterstützt	Unterstützt
Dynamische Regeln	Nicht unterstützt	Unterstützt

Überlegungen

Beachten Sie die folgenden Punkte, bevor Sie AWS Glue Data Quality verwenden:

- Datenqualitätsregeln können keine verschachtelten oder Listentyp-Datenquellen auswerten. Siehe Verschachtelte Strukturen verflachen.

Terminologie

In der folgenden Liste werden Begriffe definiert, die sich auf die AWS Glue Datenqualität beziehen.

Definitionssprache für Datenqualität (DQDL)

> Eine domänenspezifische Sprache, mit der Sie AWS Glue Datenqualitätsregeln schreiben können.

Weitere Informationen zu DQDL finden Sie im Referenz zu Data Quality Definition Language (DQDL)-Benutzerhandbuch.

Datenqualität

Beschreibt, wie gut ein Datensatz seinen spezifischen Zweck erfüllt. AWS Glue Data Quality bewertet Regeln anhand eines Datensatzes, um die Datenqualität zu messen. Jede Regel prüft auf bestimmte Merkmale wie Datenaktualität oder -integrität. Zur Quantifizierung der Datenqualität können Sie einen Datenqualitätswert verwenden.

Datenqualitätswert

Der Prozentsatz der Datenqualitätsregeln, die erfüllt werden (das Ergebnis ist wahr), wenn Sie einen Regelsatz mit AWS Glue Data Quality auswerten.

Regel

Ein DQDL-Ausdruck, der Ihre Daten auf ein bestimmtes Merkmal überprüft und einen booleschen Wert zurückgibt. Weitere Informationen finden Sie unter Regelstruktur.

Analysator

Ein DQDL-Ausdruck, der Datenstatistiken sammelt. Ein Analysator sammelt Datenstatistiken, die von ML-Algorithmen verwendet werden können, um Anomalien und hard-to-detect Datenqualitätsprobleme im Laufe der Zeit zu erkennen.

Regelsatz

Eine AWS Glue Ressource, die eine Reihe von Datenqualitätsregeln umfasst. Ein Regelsatz muss einer Tabelle im AWS Glue Data Catalog zugeordnet sein. Beim Speichern eines Regelsatzes weist AWS Glue dem Regelsatz einen Amazon-Ressourcennamen (ARN) zu.

Datenqualitätswert

Der Prozentsatz der Datenqualitätsregeln, die bei der Auswertung eines Regelsatzes mit AWS Glue Data Quality erfolgreich sind (das Ergebnis ist wahr).

Beobachtung

Eine von AWS Glue generierte unbestätigte Erkenntnis, die durch die Analyse von Datenstatistiken gewonnen wird, die im Laufe der Zeit anhand von Regeln und Analysatoren erfasst wurden.

Einschränkungen

AWS Glue Einschränkungen des Datenqualitätsdienstes:

- Ein Regelsatz kann 2.000 Regeln enthalten. Wenn Ihre Regelsätze größer sind, empfehlen wir, sie in mehrere Regelsätze aufzuteilen.
- Die Größe des Regelsatzes beträgt 65 KB. Wenn Ihre Regelsätze größer sind, empfehlen wir, sie in mehrere Regelsätze aufzuteilen.
- AWS Glue Data Quality sammelt Statistiken, wenn Sie eine Regel oder einen Analysator erstellen. Das Speichern dieser Statistiken ist mit keinen Kosten verbunden. Es gibt jedoch ein Limit von 100.000 Statistiken pro Konto, und diese Statistiken werden maximal zwei Jahre lang aufbewahrt.

Versionshinweise für AWS Glue Datenqualität

In diesem Thema werden die in AWS Glue Data Quality eingeführten Funktionen beschrieben.

Allgemeine Verfügbarkeit: neue Features

Die folgenden neuen Funktionen sind mit der allgemeinen Verfügbarkeit von AWS Glue Data Quality verfügbar:

- Die Möglichkeit, festzustellen, welche Datensätze die Datenqualitätsprüfungen nicht bestanden haben, wird jetzt unterstützt in AWS Glue Studio
- Neue Regeltypen für die Datenqualität, wie z. B. die Validierung der referenziellen Integrität von Daten zwischen zwei Datensätzen, der Vergleich von Daten zwischen zwei Datensätzen und Datentypprüfungen
- Verbesserte Benutzererfahrung in der AWS Glue Data Catalog
- Unterstützung für Apache Iceberg, Apache Hudi und Delta Lake
- Unterstützung für Amazon Redshift
- Vereinfachte Benachrichtigung mit Amazon EventBridge
- AWS CloudFormation Unterstützung für die Erstellung von Regelsätzen
- Leistungsverbesserungen: Caching-Option in ETL und AWS Glue Studio für eine schnellere Leistung bei der Bewertung der Datenqualität

27. November 2023 (Vorschau)

- ML-gestützte Funktionen zur Erkennung von Anomalien sind jetzt in AWS Glue ETL und AWS Glue Studio verfügbar. Damit können Sie jetzt Anomalien und hard-to-detect Datenqualitätsprobleme erkennen
- Mit Dynamic Rules können Sie dynamische Schwellenwerte angeben (z. B.:) `RowCount > avg(last(10))`

12. März 2024

- DQDL-Verbesserungen
  - Support für Schlüsselwörter wie NULL, BLANKS, WHITESPACES_ONLY
  - Optionen zur Angabe, wie Data Quality mit zusammengesetzten Regeln umgehen AWS Glue muss
  - ColumnValues Der Regeltyp lässt bei Vergleichen nicht zu, dass NULL-Werte übergeben werden
  - Support für den NOT-Operator in DQDL

26. Juni 2024

- DQDL-Verbesserungen
  - DQDL unterstützt jetzt die WHERE-Klausel, sodass Sie Daten filtern können, bevor Sie DQ-Regeln anwenden

7. August 2024

- Anomalieerkennung und dynamische Regeln sind jetzt allgemein verfügbar

22. November 2024

- Mit komplexen zusammengesetzten Regeln können Sie komplexere Geschäftsregeln mit geschachtelter Unterstützung erstellen
- Neue Regeltypen zur Verwaltung der Datenqualität Ihrer Dateien
  - FileFreshness

- FileSize
- FileUniqueness
- FileMatch

- Standardmäßige Datenqualitätsprüfungen in Visual ETL-Jobs

6. Dezember 2024

- AWS Glue Data Quality unterstützt jetzt Amazon SageMaker AI LakeHouse Tabellen und AWS Lake Formation verwaltete Iceberg-, Delta- und HUDI-Tabellen in AWS Glue ETL 5.0.

30. Juni 2025

- Aggregierte Metriken — jetzt können Sie aggregierte Metriken abrufen, z. B. die Anzahl der Datensätze, die auf API-Ebene bestanden wurden oder fehlgeschlagen sind, für AWS Glue ETL-basierte DQ-Jobs.

Erkennung von Anomalien in AWS Glue Data Quality

Argumente für Machine Learning im Bereich Datenqualität

Ingenieure verwalten Hunderte von Daten-Pipelines gleichzeitig. Jede Pipeline kann Daten aus verschiedenen Quellen extrahieren und in den Data Lake oder andere Datenrepositorien laden. Um sicherzustellen, dass qualitativ hochwertige Daten für die Entscheidungsfindung bereitgestellt werden, legen sie Datenqualitätsregeln fest. Diese Regeln bewerten die Daten auf der Grundlage fester Kriterien, die den aktuellen Geschäftsstand widerspiegeln. Wenn sich das Geschäftsumfeld jedoch ändert, ändern sich die Dateneigenschaften, wodurch diese festen Kriterien überholt sind und die Datenqualität beeinträchtigt wird.

Beispielsweise hat ein Dateningenieur in einem Einzelhandelsunternehmen eine Regel festgelegt, nach der der Tagesumsatz einen bestimmten one-million-dollar Schwellenwert überschreiten muss. Nach einigen Monaten überstieg der Tagesumsatz zwei Millionen Dollar, sodass der Schwellenwert nicht mehr gültig war. Der Dateningenieur konnte die Regeln nicht aktualisieren, um die aktuellen Schwellenwerte widerzuspiegeln, da keine Benachrichtigung erfolgte und die Regel aufwändig manuell analysiert und aktualisiert werden musste. Später im Monat verzeichneten Geschäftsanwender einen Umsatzrückgang von 25%. Nach stundenlangen Untersuchungen stellten die Dateningenieure fest, dass eine ETL-Pipeline, die für das Extrahieren von Daten aus einigen

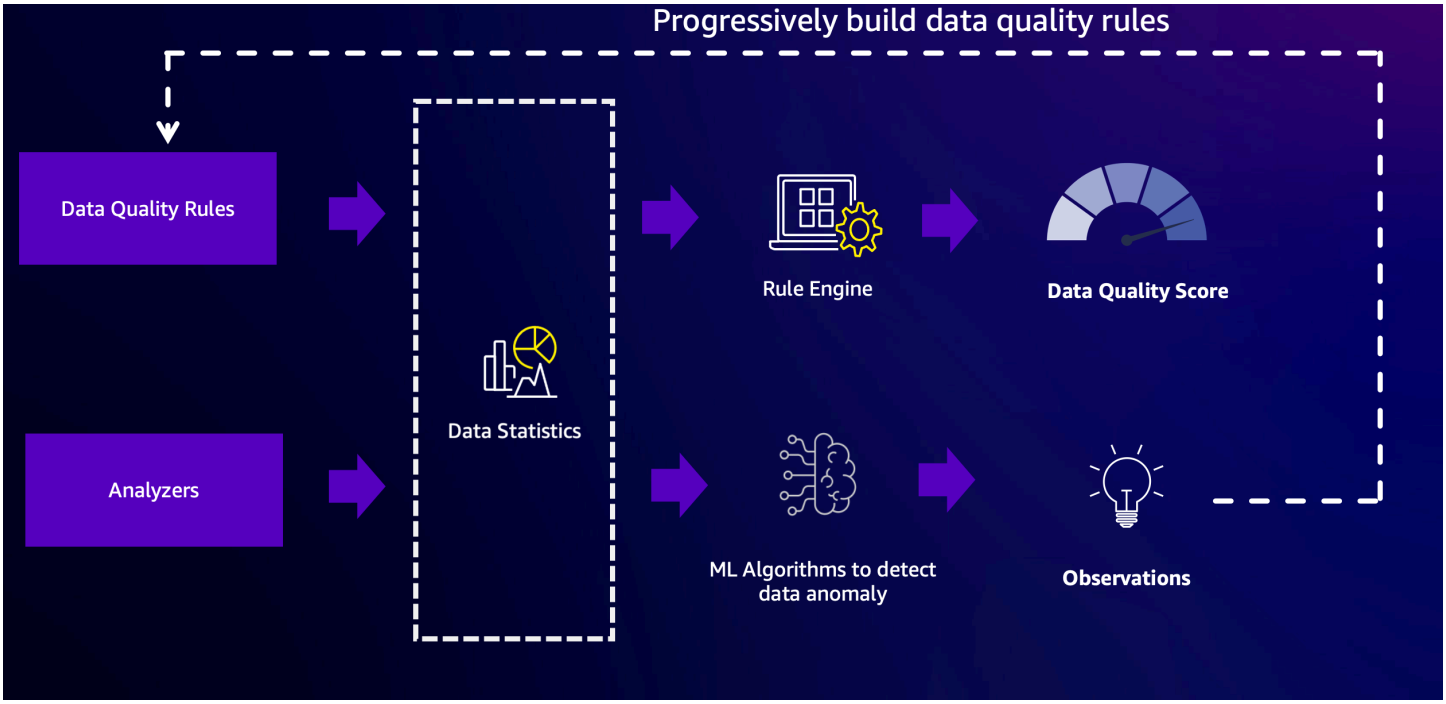
Geschäften zuständig war, ausgefallen war, ohne dass Fehler aufgetreten sind. Die Regel mit veralteten Schwellenwerten funktionierte weiterhin erfolgreich, ohne dass dieses Problem erkannt wurde.

Alternativ hätten proaktive Warnmeldungen, die diese Anomalien erkennen können, es Benutzern ermöglichen, dieses Problem zu erkennen. Darüber hinaus kann die Überwachung der Saisonalität im Geschäft auf erhebliche Probleme mit der Datenqualität hinweisen. Beispielsweise können die Einzelhandelsumsätze an Wochenenden und in der Weihnachtszeit am höchsten sein, während sie an Wochentagen relativ niedrig sind. Eine Abweichung von diesem Muster kann auf Probleme mit der Datenqualität oder Veränderungen der Geschäftsumstände hinweisen. Mit Datenqualitätsregeln können saisonale Muster nicht erkannt werden, da hierfür fortschrittliche Algorithmen erforderlich sind, die aus vergangenen Mustern lernen und saisonale Schwankungen erfassen können, um Abweichungen zu erkennen.

Und schließlich ist es für Benutzer schwierig, Regeln zu erstellen und zu verwalten, da der Regelerstellungsprozess technisch ist und viel Zeit für deren Erstellung benötigt wird. Aus diesem Grund ziehen sie es vor, zuerst Dateneinblicke zu untersuchen, bevor sie Regeln definieren. Kunden müssen in der Lage sein, Anomalien mühelos zu erkennen, sodass sie Datenqualitätsprobleme proaktiv erkennen und fundierte Geschäftsentscheidungen treffen können.

Funktionsweise

> **Note**
>
> Die Erkennung von Anomalien wird nur in AWS Glue ETL unterstützt. Dies wird bei der Datenqualität auf Basis von Data Catalog nicht unterstützt.



AWS Glue Data Quality kombiniert die Leistungsfähigkeit regelbasierter Datenqualitäts- und Anomalieerkennungsfunktionen, um qualitativ hochwertige Daten zu liefern. Um loszulegen, müssen Sie zuerst Regeln und Analysatoren konfigurieren und dann die Anomalieerkennung aktivieren.

Regeln

Regeln — Regeln drücken die Erwartungen an Ihre Daten in einer offenen Sprache aus, der sogenannten Data Quality Definition Language (DQDL). Ein Beispiel für eine Regel ist unten dargestellt. Diese Regel ist erfolgreich, wenn die Spalte `passenger_count` keine leeren Werte oder NULL-Werte enthält:

```
Rules = [
    IsComplete "passenger_count"
]
```

Analysatoren

In Situationen, in denen Sie die kritischen Spalten kennen, aber möglicherweise nicht genug über die Daten wissen, um spezifische Regeln zu schreiben, können Sie diese Spalten mit Analyzern überwachen. Analyzer sind eine Möglichkeit, Datenstatistiken zu sammeln, ohne explizite Regeln zu definieren. Ein Beispiel für die Konfiguration von Analyzern ist unten dargestellt:

```
Analyzers = [
```

```
AllStatistics "fare_amount",
DistinctValuesCount "pulocationid",
RowCount
]
```

In diesem Beispiel sind drei Analyzer konfiguriert:

1. Der erste Analyzer, `AllStatistics „fare_amount"`, erfasst alle verfügbaren Statistiken für das Feld `fare_amount`.
2. Der zweite Analyzer, `DistinctValuesCount „pulocationid"`, erfasst die Anzahl der unterschiedlichen Werte in der Spalte `pulocationid`.
3. Der dritte Analyzer, `RowCount`, erfasst die Gesamtzahl der Datensätze im Datensatz.

Analysatoren dienen als einfache Methode zur Erfassung relevanter Datenstatistiken, ohne dass komplexe Regeln festgelegt werden müssen. Durch die Überwachung dieser Statistiken können Sie Einblicke in die Datenqualität gewinnen und potenzielle Probleme oder Anomalien identifizieren, die möglicherweise weitere Untersuchungen oder die Erstellung spezifischer Regeln erfordern.

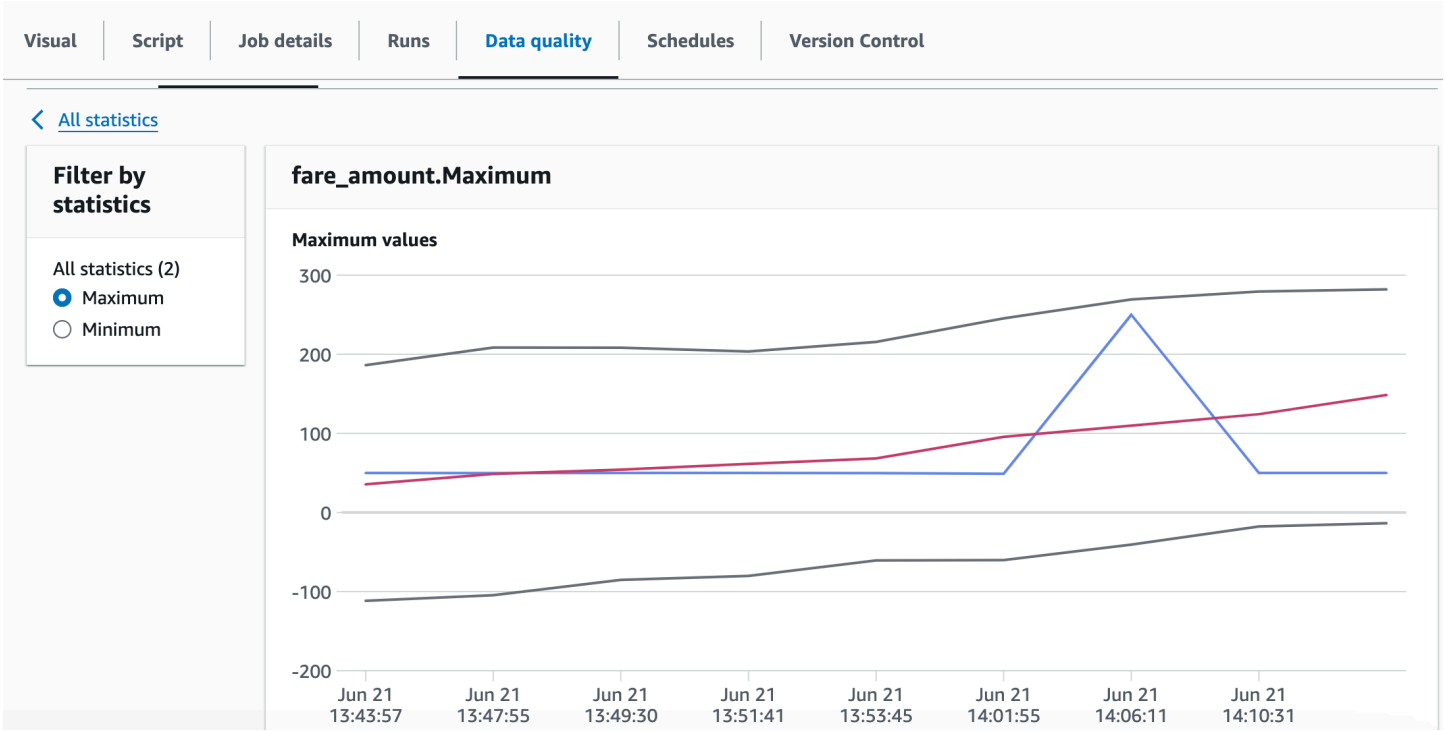
Datenstatistik

Sowohl Analyzer als auch Rules in AWS Glue Data Quality erfassen Datenstatistiken, auch bekannt als Datenprofile. Diese Statistiken bieten Einblicke in die Eigenschaften und die Qualität Ihrer Daten. Die gesammelten Statistiken werden im Laufe der Zeit innerhalb des AWS Glue-Dienstes gespeichert, sodass Sie Änderungen in Ihren Datenprofilen verfolgen und analysieren können.

Sie können diese Statistiken einfach abrufen und zur weiteren Analyse oder Langzeitspeicherung in Amazon S3 schreiben, indem Sie die entsprechende APIs Option aufrufen. Diese Funktion ermöglicht es Ihnen, die Erstellung von Datenprofilen in Ihre Datenverarbeitungsabläufe zu integrieren und die gesammelten Statistiken für verschiedene Zwecke zu nutzen, z. B. für die Überwachung der Datenqualität und die Erkennung von Anomalien.

Durch das Speichern der Datenprofile in Amazon S3 können Sie die Skalierbarkeit, Haltbarkeit und Wirtschaftlichkeit des Objektspeicherservices von Amazon nutzen. Darüber hinaus können Sie andere AWS Dienste oder Tools von Drittanbietern nutzen, um die Datenprofile zu analysieren und zu visualisieren, sodass Sie tiefere Einblicke in Ihre Datenqualität gewinnen und fundierte Entscheidungen über Datenmanagement und Datenverwaltung treffen können.

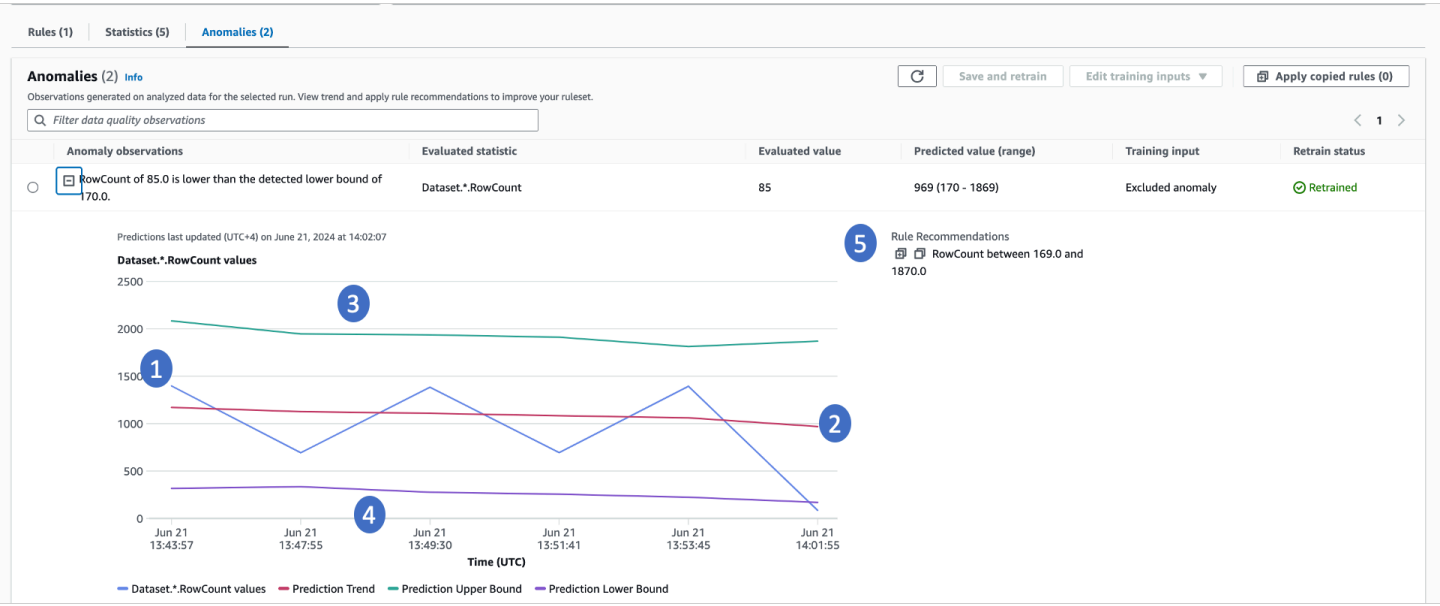
Hier ist ein Beispiel für Datenstatistiken, die im Laufe der Zeit gespeichert wurden.



> **Note**
>
> AWS Glue Data Quality erfasst Statistiken nur einmal, auch wenn Sie sowohl Rule als auch Analyzer für dieselben Spalten haben, wodurch der Prozess der Statistikgenerierung effizient wird.

Anomalieerkennung

AWS Glue Data Quality erfordert mindestens drei Datenpunkte, um Anomalien zu erkennen. Es verwendet einen Algorithmus für maschinelles Lernen, um aus vergangenen Trends zu lernen und dann future Werte vorherzusagen. Wenn der tatsächliche Wert nicht innerhalb des vorhergesagten Bereichs liegt, erstellt AWS Glue Data Quality eine Anomaliebeobachtung. Es bietet eine visuelle Darstellung des tatsächlichen Werts und der Trends. In der folgenden Grafik werden vier Werte angezeigt.



1. Die aktuelle Statistik und ihr Trend im Zeitverlauf.
2. Ein abgeleiteter Trend, der durch Lernen aus dem tatsächlichen Trend abgeleitet wird. Dies ist nützlich, um die Trendrichtung zu verstehen.
3. Die mögliche Obergrenze für die Statistik.
4. Die mögliche Untergrenze für die Statistik.
5. Empfohlene Datenqualitätsregeln, mit denen diese Probleme in future erkannt werden können.

In Bezug auf Anomalien sind einige wichtige Dinge zu beachten:

- Wenn Anomalien generiert werden, hat dies keinen Einfluss auf die Datenqualitätswerte.
- Wenn eine Anomalie erkannt wird, wird sie bei nachfolgenden Durchläufen als normal angesehen. Der Algorithmus für maschinelles Lernen betrachtet diesen anomalen Wert als Eingabe, sofern er nicht ausdrücklich ausgeschlossen wird.

Umschulung

Eine Umschulung des Modells zur Erkennung von Anomalien ist entscheidend, um die richtigen Anomalien zu erkennen. Wenn Anomalien erkannt werden, nimmt AWS Glue Data Quality die Anomalie als Normalwert in das Modell auf. Um sicherzustellen, dass die Anomalieerkennung korrekt funktioniert, ist es wichtig, Feedback zu geben, indem die Anomalie bestätigt oder abgelehnt wird. AWS Glue Data Quality bietet Mechanismen sowohl in AWS Glue Studio als auch APIs zur

Bereitstellung von Feedback zum Modell. Weitere Informationen finden Sie in der Dokumentation zur Einrichtung der Anomalieerkennung in AWS Glue-ETL-Pipelines.

Einzelheiten zum Algorithmus zur Erkennung von Anomalien

- Der Algorithmus zur Erkennung von Anomalien untersucht Datenstatistiken im Zeitverlauf. Der Algorithmus berücksichtigt alle verfügbaren Datenpunkte und ignoriert alle Statistiken, die explizit ausgeschlossen wurden.
- Diese Datenstatistiken werden im AWS Glue-Dienst gespeichert, und Sie können AWS KMS Schlüssel zur Verschlüsselung bereitstellen. Informationen zur Bereitstellung von AWS KMS Schlüsseln zur Verschlüsselung der AWS Glue-Datenqualitätsstatistiken finden Sie im Sicherheitsleitfaden.
- Die Zeitkomponente ist entscheidend für den Algorithmus zur Erkennung von Anomalien. Basierend auf vergangenen Werten bestimmt AWS Glue Data Quality die Ober- und Untergrenzen. Bei dieser Bestimmung wird die Zeitkomponente berücksichtigt. Die Grenzwerte unterscheiden sich für dieselben Werte in einem Intervall von einer Minute, einem Stundenintervall oder einem Tagesintervall.

Erfassung der Saisonalität

AWS Der Algorithmus zur Erkennung von Anomalien von Glue Data Quality kann saisonale Muster erfassen. Es kann beispielsweise nachvollziehen, dass sich die Muster an Wochentagen von den Mustern am Wochenende unterscheiden. Dies ist im folgenden Beispiel zu sehen, in dem AWS Glue Data Quality einen saisonalen Trend in den Datenwerten erkennt. Sie müssen nichts Spezielles tun, um diese Funktion zu aktivieren. Im Laufe der Zeit lernt AWS Glue Data Quality saisonale Trends kennen und erkennt Anomalien, wenn diese Muster durchbrechen.

Kosten

Die Gebühr richtet sich nach der Zeit, die für die Erkennung von Anomalien benötigt wird. Für jede Statistik wird 1 DPU für die Zeit berechnet, die zur Erkennung von Anomalien benötigt wird. Ausführliche Beispiele finden Sie unter AWS Glue Pricing.

Die wichtigsten Überlegungen

Die Speicherung der Statistiken ist kostenlos. Es gibt jedoch ein Limit von 100.000 Statistiken pro Konto. Diese Statistiken werden für maximal zwei Jahre gespeichert.

Konfigurieren Sie IAM-Berechtigungen für AWS Glue Data Quality

Dieses Thema enthält Informationen, die Ihnen helfen sollen, die Aktionen und Ressourcen zu verstehen, die Sie als IAM-Administrator in einer AWS Identity and Access Management (IAM-) Richtlinie für AWS Glue Data Quality verwenden können. Es enthält auch Beispiel-IAM-Richtlinien mit den Mindestberechtigungen, die Sie benötigen, um AWS Glue Data Quality mit dem AWS Glue Data Catalog zu verwenden.

Weitere Informationen zur Sicherheit in AWS Glue finden Sie unterSicherheit in AWS Glue.

IAM-Berechtigungen für AWS Glue Data Quality

Die folgende Tabelle listet die Berechtigungen auf, die ein Benutzer benötigt, um bestimmte Vorgänge für AWS Glue Data Quality auszuführen. Um eine detaillierte Autorisierung für AWS Glue Data Quality festzulegen, können Sie diese Aktionen im `Action` Element einer IAM-Richtlinienerklärung angeben.

AWS Maßnahmen zur Datenqualität mit Glue

Aktion	Beschreibung	Ressourcentypen
`glue:CreateDataQualityRuleset`	Gewährt die Berechtigung zum Erstellen eines Datenqualitätsregelsatzes.	`::dataQualityRuleset/<name>`
`glue:DeleteDataQualityRuleset`	Gewährt die Berechtigung zum Löschen eines Datenqualitätsregelsatzes.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityRuleset`	Gewährt die Berechtigung zum Abrufen eines Datenqualitätsregelsatzes.	`::dataQualityRuleset/<name>`
`glue:ListDataQualityRulesets`	Gewährt die Berechtigung zum Abrufen aller Datenqualitätsregelsätze.	`::dataQualityRuleset/*`
`glue:UpdateDataQualityRuleset`	Gewährt die Berechtigung zum Aktualisieren eines Datenqualitätsregelsatzes.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityResult`	Gewährt die Berechtigung zum Abrufen eines Ausführungsergebnisses einer Datenqualitätsaufgabe. Diese IAM-Aktion bietet auch Berechtigungen für die folgenden APIS:	`::dataQualityRuleset/<name>`

Aktion	Beschreibung	Ressourcentypen
	• BatchGetDataQualityQualityResult • ListDataQualityStatistics • ListDataQualityStatisticAnnotations	
glue:ListDataQualityResults	Gewährt die Berechtigung zum Abrufen aller Ausführungsergebnisse einer Datenqualitätsaufgabe.	::dataQualityRuleset/*
glue:CancelDataQualityRuleRecommendationRun	Gewährt die Berechtigung, die Ausführung einer gerade ausgeführten Aufgabe zur Datenqualitätsempfehlung anzuhalten.	::dataQualityRuleset/*
glue:GetDataQualityRuleRecommendationRun	Gewährt die Berechtigung zum Abrufen einer Ausführung einer Aufgabe zur Datenqualitätsempfehlung.	::dataQualityRuleset/*
glue:ListDataQualityRuleRecommendationRuns	Gewährt die Berechtigung zum Abrufen aller Ausführungen der Aufgaben zur Datenqualitätsempfehlung.	::dataQualityRuleset/*
glue:StartDataQualityRuleRecommendationRun	Gewährt die Berechtigung zum Starten einer Aufgabe zur Datenqualitätsempfehlung.	::dataQualityRuleset/*
glue:CancelDataQualityRulesetEvaluationRun	Gewährt die Berechtigung zum Anhalten einer gerade ausgeführten Datenqualitätsaufgabe.	::dataQualityRuleset/*

Aktion	Beschreibung	Ressourcentypen
glue:GetDataQualityRulesetEvaluationRun	Gewährt die Berechtigung zum Abrufen einer ausgeführten Datenqualitätsaufgabe.	::dataQualityRuleset/*
glue:ListDataQualityRulesetEvaluationRuns	Gewährt die Berechtigung zum Abrufen aller Ausführungen von Datenqualitätsaufgaben.	::dataQualityRuleset/*
glue:StartDataQualityRulesetEvaluationRun	Gewährt die Berechtigung zum Starten der Ausführung einer Datenqualitätsaufgabe.	::dataQualityRuleset/<name>
glue:PublishDataQuality	Erteilt die Genehmigung zur Veröffentlichung von Datenqualitätsergebnissen.	::dataQualityRuleset/<name>
glue:GetDataQualityModel	Erteilt die Berechtigung zum Abrufen des Datenqualitätsmodells.	::dataQualityRuleset/<name>, ::job/<name>
glue:GetDataQualityModelResult	Erteilt die Berechtigung zum Abrufen der Ergebnisse des Datenqualitätsmodells.	::dataQualityRuleset/<name>, ::job/<name>
glue:PutDataQualityStatisticAnnotation	Erteilt die Berechtigung zum Hinzufügen von Anmerkungen zu Statistics. Diese IAM-Aktion bietet auch Berechtigungen für die folgenden APIS: • BatchPutDataQualityStatisticAnnotation	::dataQualityRuleset/<name>, ::job/<name>

Aktion	Beschreibung	Ressourcentypen
`glue:PutDataQualityProfileAnnotation`	Erteilt die Berechtigung, Anmerkungen zu allen Statistiken in einem Profil hinzuzufügen.	`::dataQualityRuleset/<name>, ::job/<name>`

Für die Planung von Auswertungsausführungen erforderliche IAM-Einrichtung

IAM-Berechtigungen

Um geplante Data-Quality-Auswertungsausführungen auszuführen, müssen Sie die `IAM:PassRole`-Aktion zur Berechtigungsrichtlinie hinzufügen.

AWS EventBridge Für den Scheduler sind Berechtigungen erforderlich

Aktion	Beschreibung	Ressourcentypen
`iam:PassRole`	Gewährt IAM die Berechtigung, dem Benutzer die Übergabe der genehmigten Rollen zu ermöglichen.	ARN der Rolle, die zum Aufrufen von `StartDataQualityRulesetEvaluationRun` verwendet wird

Ohne diese Berechtigungen tritt der folgende Fehler auf:

```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not
authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-
role/AWSGlueServiceRole
because no identity-based policy allows the iam:PassRole action"
```

Vertrauenswürdige IAM-Entitäten

Die Dienste AWS Glue und AWS EventBridge Scheduler müssen in den vertrauenswürdigen Entitäten aufgeführt sein, um einen Zeitplan `StartDataQualityEvaluationRun` zu erstellen und auszuführen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

IAM-Beispielrichtlinien

Eine IAM-Rolle für AWS Glue Data Quality benötigt die folgenden Arten von Berechtigungen:

- Berechtigungen für AWS Glue-Datenqualitätsoperationen, sodass Sie empfohlene Datenqualitätsregeln abrufen und eine Datenqualitätsaufgabe anhand einer Tabelle im AWS Glue-Datenkatalog ausführen können. Die IAM-Beispielrichtlinien in diesem Abschnitt enthalten die Mindestberechtigungen, die für AWS Glue Data Quality-Operationen erforderlich sind.
- Berechtigungen, die Zugriff auf Ihre Datenkatalogtabelle und die zugrunde liegenden Daten gewähren. Diese Berechtigungen variieren je nach Anwendungsfall. Beispielsweise sollten die Berechtigungen für Daten, die Sie in Amazon S3 katalogisieren, den Zugriff auf Amazon S3 beinhalten.

> **Note**
>
> Sie müssen zusätzlich zu den in diesem Abschnitt beschriebenen Berechtigungen Amazon-S3-Berechtigungen konfigurieren.

Mindestberechtigungen zum Abrufen empfohlener Datenqualitätsregeln

Diese Beispielrichtlinie enthält die Berechtigungen, die Sie benötigen, um empfohlene Datenqualitätsregeln zu generieren.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGlueRuleRecommendationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleRecommendationRun",
        "glue:PublishDataQuality",
        "glue:CreateDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
 {
     "Sid": "AllowCatalogPermissions",
     "Effect": "Allow",
     "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "AllowS3GetObjectToRunRuleRecommendationTask",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
```

```
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
 {   // Optional for Logs
        "Sid": "AllowPublishingCloudwatchLogs",
        "Effect": "Allow",
        "Action": [
          "logs:CreateLogStream",
          "logs:CreateLogGroup",
          "logs:PutLogEvents"
        ],
        "Resource": "*"
    },
  ]
}
```

Gewährt die Berechtigung zum Starten der Ausführung einer Datenqualitätsaufgabe

Diese Beispielrichtlinie enthält die Berechtigungen, die Sie benötigen, um eine Aufgabe zur Bewertung der Datenqualität auszuführen.

Die folgenden Richtlinienerklärungen sind optional, je nach Anwendungsfall:

- `AllowCloudWatchPutMetricDataToPublishTaskMetrics` - Erforderlich, wenn Sie Datenqualitäts-Ausführungsmetriken auf Amazon CloudWatch veröffentlichen wollen.
- `AllowS3PutObjectToWriteTaskResults` - Erforderlich, wenn Sie die Ergebnisse von Datenqualitätsläufen in Amazon S3 schreiben möchten.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGlueGetDataQualityRuleset",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/<YOUR-RULESET-NAME>"
    },
```

```
    {
      "Sid": "AllowGlueRulesetEvaluationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRulesetEvaluationRun",
        "glue:PublishDataQuality"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
  {
      "Sid": "AllowCatalogPermissions",
      "Effect": "Allow",
      "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "AllowS3GetObjectForRulesetEvaluationRun",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
    {
      "Sid": "AllowCloudWatchPutMetricDataToPublishTaskMetrics",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:PutMetricData"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloudwatch:namespace": "Glue Data Quality"
        }
      }
    },
    {
      "Sid": "AllowS3PutObjectToWriteTaskResults",
      "Effect": "Allow",
```

```
      "Action": [
        "s3:PutObject*"
      ],
      "Resource": "arn:aws:s3:::<YOUR-BUCKET-NAME>/*"
    }
  ]
}
```

Mindestberechtigungen für die Ausführung eines ETL-Jobs mit Datenqualität

Diese Beispielrichtlinie enthält die Berechtigungen, die Sie benötigen, um einen ETL-Job für Datenqualität auszuführen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGluePublishDataQualityResult",
      "Effect": "Allow",
      "Action": [
        "glue:PublishDataQuality"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
    //Optional to retrieve results, observation generation,
    //dynamic rules and DetectAnomalies
    {
      "Sid": "AllowGlueGetDataQualityResult",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityResult"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
    //Optional to allow annotating statistics
    {
      "Sid": "AllowGlueDataQualityStatisticAnnotation",
      "Effect": "Allow",
      "Action": [
        "glue:PutDataQualityStatisticAnnotation"
      ],
      "Resource": [
```

```
        "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*",
        "arn:aws:glue:us-east-1:111122223333::job/{JobName}"
      ]
    },
    //Optional to allow annotating all statistics in a profile
    {
      "Sid": "AllowGlueDataQualityProfileAnnotation",
      "Effect": "Allow",
      "Action": [
        "glue:PutDataQualityProfileAnnotation"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*",
        "arn:aws:glue:us-east-1:111122223333::job/{JobName}"
      ]
    }
}
```

Erste Schritte mit AWS Glue Data Quality für die Data Catalog

Dieser Abschnitt „Erste Schritte“ enthält Anweisungen, die Ihnen bei den ersten Schritten helfen AWS Glue Data Quality auf dem AWS Glue console. Sie lernen, wie Sie grundlegende Aufgaben wie das Generieren von Empfehlungen für Datenqualitätsregeln und das Auswerten eines Regelsatzes anhand Ihrer Daten ausführen.

Themen

- Voraussetzungen
- Step-by-step Beispiel
- Generieren von Regelempfehlungen
- Empfehlungen zu Überwachungsregeln
- Bearbeitung von empfohlenen Regelsätzen
- Erstellen eines neuen Regelsatzes
- Ausführen eines Regelsatzes zur Bewertung der Datenqualität
- Aufrufen des Datenqualitätswerts und der Ergebnisse
- Verwandte Themen

Voraussetzungen

Bevor du benutzt AWS Glue Data Quality, Sie sollten mit der Verwendung des vertraut sein Data Catalog und Crawler rein AWS Glue. Mit AWS Glue Data Quality, Sie können die Qualität von Tabellen in einem bewerten Data Catalog Datenbank. Sie benötigen außerdem Folgendes:

- Eine Tabelle in der Data Catalog um Ihren Regelsatz für die Datenqualität anhand zu bewerten.
- Eine IAM-Rolle für AWS Glue die Sie angeben, wenn Sie Regelempfehlungen generieren oder eine Datenqualitätsaufgabe ausführen. Diese Rolle muss berechtigt sein, auf verschiedene Ressourcen zuzugreifen AWS Glue Data Quality Prozesse müssen in Ihrem Namen ausgeführt werden. Zu diesen Ressourcen gehören AWS Glue, Amazon S3 und CloudWatch. Um Beispielrichtlinien anzuzeigen, die die Mindestberechtigungen für enthalten AWS Glue Data Quality, finden Sie unter IAM-Beispielrichtlinien.

  Um mehr über IAM-Rollen zu erfahren für AWS Glue, siehe Erstellen einer IAM-Richtlinie für AWS Glue service und Erstellen Sie eine IAM-Rolle für AWS Glue Dienst. Sie können sich auch eine Liste aller ansehen AWS Glue Berechtigungen, die spezifisch für die Datenqualität in Authorization for sind AWS Glue Data Quality Aktionen.
- Eine Datenbank mit mindestens einer Tabelle, die verschiedene Daten enthält. Die in diesem Tutorial verwendete Tabelle trägt den Namen `yyz-tickets` und die Tabelle `tickets`. Bei diesen Daten handelt es sich um eine Sammlung öffentlich zugänglicher Informationen der Stadt Toronto zu Parkplatzgebühren. Wenn Sie Ihre eigene Tabelle erstellen, stellen Sie sicher, dass diese mit einer Vielzahl gültiger Daten ausgefüllt ist, um die besten empfohlenen Regeln zu erhalten.

Step-by-step Beispiel

Ein step-by-step Beispiel mit Beispieldatensätzen finden Sie im Blogbeitrag AWS Glue Data Quality.

Generieren von Regelempfehlungen

Regelempfehlungen vereinfachen den Einstieg in die Datenqualität, ohne Code schreiben zu müssen. Mit AWS Glue Data Quality können Sie Ihre Daten analysieren, Regeln identifizieren und einen Regelsatz erstellen, den Sie in einer Datenqualitätsaufgabe auswerten können. Empfehlungsausführungen werden nach 90 Tagen automatisch gelöscht.

So generieren Sie Empfehlungen für Datenqualitätsregeln

1. Öffnen Sie die AWS Glue-Konsole unter https://console.aws.amazon.com/glue/.

2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus. Wählen Sie anschließend die Tabelle aus, für die Sie Empfehlungen für Datenqualitätsregeln generieren möchten.
3. Wählen Sie auf der Seite mit den Tabellendetails die Registerkarte Datenqualität aus, um auf die AWS Glue-Datenqualitätsregeln und -einstellungen für Ihre Tabelle zuzugreifen.
4. Wählen Sie auf der Registerkarte Datenqualität die Option Regeln hinzufügen und Datenqualität überwachen aus.
5. Auf der Seite Regelsatz-Generator werden Sie durch eine Warnung oben auf der Seite aufgefordert, eine Empfehlungsaufgabe zu starten, wenn keine Regelempfehlungen ausgeführt werden.
6. Wählen Sie Regeln empfehlen, um das Modal zu öffnen und Ihre Parameter für die Empfehlungsaufgabe einzugeben.
7. Wählen Sie eine IAM-Rolle mit Zugriff auf AWS Glue. Diese Rolle muss berechtigt sein, auf Ressourcen zuzugreifen, die verschiedene AWS Glue Data Quality-Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden.
8. Nachdem die Felder gemäß Ihren Präferenzen ausgefüllt wurden, wählen Sie Regeln empfehlen, um die Ausführung der Empfehlungsaufgabe zu starten. Wenn Empfehlungsausführungen in Arbeit oder abgeschlossen sind, können Sie Ihre Ausführungen in dieser Benachrichtigung verwalten. Möglicherweise müssen Sie die Warnmeldung aktualisieren, um die Statusänderung anzuzeigen. Abgeschlossene und laufende Ausführungen von Empfehlungsaufgaben werden auf der Seite Ausführungsverlauf angezeigt, die alle Empfehlungsausführungen in den vergangenen 90 Tagen auflistet.

Bedeutung der empfohlenen Regeln

AWS Glue Data Quality generiert Regeln auf der Grundlage der Daten aus jeder Spalte der Eingabetabelle. Mithilfe der Regeln werden mögliche Grenzen identifiziert, an denen Daten gefiltert werden können, um Qualitätsanforderungen einzuhalten. Die folgende Liste generierter Regeln enthält Beispiele, die hilfreich sind, um zu verstehen, was die Regeln bedeuten und was sie bewirken können, wenn sie auf Ihre Daten angewendet werden.

Eine vollständige Liste der generierten DQDL-Regeltypen (Data Quality Definition Language) finden Sie in der DQDL-Regeltypreferenz.

- `IsComplete "SET_FINE_AMOUNT"` –Die `IsComplete`-Regel überprüft, ob die Spalte für eine bestimmte Zeile ausgefüllt ist. Verwenden Sie diese Regel, um Spalten in Daten als nicht optional zu kennzeichnen.

- `Uniqueness "TICKET_NUMBER" > 0.95` – Die `Uniqueness`-Regel überprüft, ob die Daten in der Spalte einen bestimmten Eindeutigkeitsschwellenwert erreichen. In diesem Beispiel wurde festgestellt, dass die Daten, die eine bestimmte Zeile für `"TICKET_NUMBER"` füllen, höchstens zu 95 % inhaltlich mit allen anderen Zeilen identisch sind, was auf diese Regel schließen lässt.
- `ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY",...]` – Die `ColumnValues`-Regel definiert gültige Werte für die Spalte, basierend auf vorhandenen Spalteninhalten. In diesem Beispiel handelt es sich bei den Daten für jede Zeile um ein aus zwei Buchstaben bestehendes Nummernschild für ein Bundesland oder eine Provinz.
- `ColumnLength "INFRACTION_DESCRIPTION" between 15 and 31` – Die `ColumnLength`-Regel erzwingt eine Längenbeschränkung der Daten einer Spalte. Diese Regel wird aus den Beispieldaten basierend auf der minimalen und maximalen aufgezeichneten Länge für eine Zeichenfolgenspalte generiert.

Empfehlungen zu Überwachungsregeln

Wenn Empfehlungen zu Datenqualitätsregeln ausgeführt werden, werden auf der Seite Regeln hinzufügen und Datenqualität überwachen in der oberen Leiste Informationen und zusätzliche Aktionen angezeigt, die Sie ausführen können.

Wenn Regelempfehlungen ausgeführt werden, können Sie die Ausführung beenden, bevor die Empfehlungsaufgabe abgeschlossen ist. Während die Aufgabe ausgeführt wird, sehen Sie den Status In Bearbeitung sowie das Datum und die Uhrzeit, zu der die Ausführung gestartet wurde.

Wenn die Regelempfehlungen abgeschlossen sind, werden in der Regelempfehlungsleiste die Anzahl der empfohlenen Regeln, der Status der letzten Empfehlungsausführung sowie das Datum und der Zeitstempel der Fertigstellung angezeigt.

Sie können die empfohlenen Regeln hinzufügen, indem Sie Regelempfehlung einfügen auswählen. Wählen Sie ein bestimmtes Datum aus, um zuvor empfohlene Regeln anzuzeigen. Um eine neue Empfehlung auszuführen, wählen Sie Weitere Aktionen und dann Empfohlene Regeln aus.

Legen Sie Standardeinstellungen fest, indem Sie Benutzereinstellungen verwalten auswählen. Sie können den Standardpfad für Amazon S3 festlegen, um Regelsätze zu speichern oder eine Standardrolle zum Ausführen des Data Catalogs einzurichten.

Bearbeitung von empfohlenen Regelsätzen

Da AWS Glue Data Quality Regeln basierend auf vorhandenen Daten generiert, die Ihnen zur Verfügung stehen, werden in den automatisierten Vorschlägen möglicherweise einige unerwartete oder unerwünschte Regeln angezeigt. Um den größtmöglichen Nutzen aus den empfohlenen Regelsätzen zu ziehen, müssen Sie diese auswerten und ändern. Für diesen Schritt des Tutorials nehmen Sie die im vorherigen Schritt generierten Regeln und passen sie an, um für einige Daten restriktivere Eigenschaften zu erzwingen. Außerdem lockern Sie andere Regeln, um sicherzustellen, dass später korrekte, eindeutige Daten hinzugefügt werden können.

Einen vorgeschlagenen Regelsatz bearbeiten

1. Wählen Sie in der AWS Glue-Konsole Datenkatalog und dann im Navigationsbereich Datenbanktabellen aus. Wählen Sie die `tickets` Tabelle aus.
2. Wählen Sie auf der Seite mit den Tabellendetails die Registerkarte Datenqualität, um auf die AWS Glue-Datenqualitätsregeln und -einstellungen für die Tabelle zuzugreifen.
3. Wählen Sie im Abschnitt Regelsätze den in Generieren von Regelempfehlungen generierten Regelsatz aus.
4. Wählen Sie Aktionen und anschließend im Konsolenfenster Bearbeiten aus. Der Regelsatz-Editor wird in der Konsole geladen. Es enthält einen Bearbeitungsbereich für Ihre Regeln und eine Kurzreferenz für DQDL.
5. Entfernen Sie die Zeile 2 des Skripts. Dadurch wird die Anforderung gelockert, dass die Datenbankgröße auf eine bestimmte Anzahl von Zeilen beschränkt sein muss. Nach der Bearbeitung sollte Ihre Datei in den Zeilen 1–3 Folgendes enthalten:

```
Rules = [
    IsComplete "TAG_NUMBER_MASKED",
    ColumnLength "TAG_NUMBER_MASKED" between 6 and 9,
```

6. Entfernen Sie die Zeile 25 des Skripts. Dies lockert die Anforderung, dass 96 % der erfassten Provinzen ON sein müssen. Nach der Bearbeitung sollte Ihre Datei von der Zeile 24 bis zum Ende des Regelsatzes Folgendes enthalten:

```
ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY", "AZ", "NS", "BC", "MI", "PQ",
 "MB", "PA", "FL", "SK", "NJ", "OH", "NB", "IL", "MA", "CA",
    "VA", "TX", "NF", "MD", "PE", "CT", "NC", "GA", "IN", "OR", "MN", "TN", "WI",
 "KY", "MO", "WA", "NH", "SC", "CO", "OK", "VT", "RI", "ME", "AL",
```

```
    "YT", "IA", "DE", "AR", "LA", "XX", "WV", "MT", "KS", "NT", "DC", "NV", "NE",
 "UT", "MS", "NM", "ID", "SD", "ND", "AK", "NU", "GO", "WY", "HI"],
ColumnLength "PROVINCE" = 2
]
```

7. Ändern Sie die Zeile 14 wie folgt:

   ```
   IsComplete "TIME_OF_INFRACTION",
   ```

   Dies verschärft die Anforderungen an die Spalte, indem die Datenbank nur auf Tickets beschränkt wird, die einen aufgezeichneten Zeitpunkt des Verstoßes enthalten. Sie sollten Tickets ohne aufgezeichneten Zeitpunkt des Verstoßes immer als ungültige Daten in diesem Datensatz betrachten. Dies unterscheidet sich von Situationen, in denen eine Partitionierung oder Transformation für die weitere Datenverwendung oder -prüfung zur Bestimmung einer Qualitätsregel besser geeignet sein könnte.

8. Wählen Sie unten auf der Konsolenseite die Option Regelsatz aktualisieren.

Erstellen eines neuen Regelsatzes

Ein Regelsatz ist eine Gruppe von Datenqualitätsregeln, die Sie anhand Ihrer Daten auswerten. Im AWS Glue In der Konsole können Sie mithilfe der Data Quality Definition Language (DQDL) benutzerdefinierte Regelsätze erstellen.

So erstellen Sie einen Datenqualitätsregelsatz

1. Wählen Sie in der AWS Glue-Konsole Datenkatalog, Datenbanken und dann Tabellen im Navigationsbereich aus. Wählen Sie die Tabelle `tickets` aus.
2. Öffnen Sie die Registerkarte Data quality (Datenqualität).
3. Wählen Sie im Abschnitt Regelsatzh die Option Regelsatz erstellen aus. Der DQDL-Editor wird in der Konsole gestartet. Es verfügt über einen Textbereich zur direkten Bearbeitung sowie eine Kurzreferenz für DQDL-Regeln und das Tabellenschema.
4. Beginnen Sie mit dem Hinzufügen von Regeln zum Textbereich des DQDL-Editors. Sie können entweder direkt aus diesem Tutorial heraus Regeln schreiben oder dieses Feature des DQDLRegelgenerators im Editor für Datenqualitätsregeln verwenden.

> **Note**
>
> Verwendung des DQDL-Regel-Generators
>
> 1. Wählen Sie einen Regeltyp aus der Liste aus und klicken Sie auf das Pluszeichen, um eine Beispielsyntax in den Editorbereich einzufügen.
> 2. Tauschen Sie die Platzhalter-Spaltennamen mit Ihren eigenen Spaltennamen aus. Spaltennamen aus der Tabelle sind auf der Registerkarte Schema verfügbar.
> 3. Aktualisieren Sie den Ausdrucksparameter nach Bedarf. Eine vollständige Liste der von DQDL unterstützten Ausdrücke finden Sie unter Ausdrücke.

Beispielsweise handelt es sich bei den folgenden Regeln um Einschränkungen für die Datenvalidierung der `ticket_number`-Spalte in der `tickets`-Tabelle. Um die folgenden Regeln hinzuzufügen, verwenden Sie den DQDL-Regelgenerator oder bearbeiten Sie Ihren Regelsatz direkt:

```
IsComplete "ticket_number",
IsUnique "ticket_number",
ColumnValues "ticket_number" > 9000000000
```

5. Geben Sie im Feld Regelsatzname einen Namen für Ihren neuen Regelsatz ein.
6. Wählen Sie Regelsatz speichern aus.

Auswertung der Datenqualität über mehrere Datensätze hinweg

Mithilfe ReferentialIntegrity von DatasetMatch Regelsätzen können Sie Datenqualitätsregeln für mehrere Datensätze einrichten. ReferentialIntegrityprüft, ob Daten im Primärdatensatz in anderen Datensätzen vorhanden sind.

Um einen Referenzdatensatz hinzuzufügen, wählen Sie die Registerkarte Schema und dann Referenztabellen aktualisieren aus. Sie werden aufgefordert, eine Datenbank und eine Tabelle auszuwählen. Sie können die Tabelle hinzufügen und dann Datenqualitätsregeln einrichten. Regeltypen wie AggregateMatch, RowCountMatch, ReferentialIntegrity SchemaMatch, und DatasetMatch unterstützen die Möglichkeit, Datenqualitätsprüfungen für mehrere Datensätze durchzuführen.

Ausführen eines Regelsatzes zur Bewertung der Datenqualität

Wenn Sie eine Datenqualitätsaufgabe ausführen, bewertet AWS Glue Data Quality einen Regelsatz anhand Ihrer Daten und berechnet einen Datenqualitätsfaktor. Dieser Wert stellt den Prozentsatz der Datenqualitätsregeln dar, die für die Eingabe bestanden haben.

So führen Sie eine Datenqualitätsaufgabe aus

1. Wählen Sie in der AWS Glue-Konsole Datenkatalog, Datenbanken und dann Tabellen im Navigationsbereich aus. Wählen Sie die Tabelle `tickets` aus.
2. Wählen Sie die Registerkarte Datenqualität.
3. Wählen Sie in der Liste Regelsätze den Regelsatz aus, den Sie anhand der Tabelle auswerten möchten. Für diesen Schritt empfehlen wir die Verwendung eines Regelsatzes, den Sie bereits geschrieben oder geändert haben, anstelle generierter Regeln. Wählen Sie Ausführen aus.
4. Wählen Sie im Modal Ihre IAM-Rolle aus. Diese Rolle muss berechtigt sein, auf Ressourcen zuzugreifen, die verschiedene AWS Glue Data Quality-Prozesse benötigen, um in Ihrem Namen ausgeführt zu werden. Sie können die IAM-Rolle als Standard speichern oder sie ändern, indem Sie die Seite mit den Standardeinstellungen aufrufen.
5. Wählen Sie unter Datenqualitätsaktionen aus, ob Sie Kennzahlen auf Amazon veröffentlichen möchten CloudWatch. Wenn diese Option ausgewählt ist, veröffentlicht AWS Glue Data Quality Metriken, die die Anzahl der Regeln angeben, die bestanden wurden, und die Anzahl der Regeln, die fehlgeschlagen sind. Um auf diese Weise gespeicherte Metriken zu bearbeiten, können Sie CloudWatch Alarme verwenden. Wichtige Kennzahlen werden ebenfalls veröffentlicht, Amazon EventBridge damit Sie Benachrichtigungen einrichten können. Weitere Informationen finden Sie unter Einrichten von Warnmeldungen, Bereitstellungen und Planung.
6. Wählen Sie unter Ausführungshäufigkeit die Option „Bei Bedarf ausführen“ oder „Regelsatz planen“ aus. Wenn Sie einen Regelsatz planen, werden Sie zur Eingabe eines Aufgabennamens aufgefordert. Der Zeitplan wird in Amazon EventBridge erstellt. Sie können Ihren Zeitplan in Amazon EventBridge bearbeiten.
7. Um die Datenqualitätsergebnisse in Amazon S3 zu speichern, wählen Sie einen Speicherort für Datenqualitätsergebnisse aus. Die IAM-Rolle, die Sie zuvor für diese Aufgabe ausgewählt haben, muss über Schreibzugriff für diesen Speicherort verfügen.
8. Geben Sie unter Zusätzliche Konfigurationen die angeforderte Anzahl von Mitarbeitern ein, die AWS Glue Ihrer Datenqualitätsaufgabe zuweisen soll.
9. Optional können Sie einen Filter an der Datenquelle einrichten. Dadurch können Sie die Menge der gelesenen Daten reduzieren. Sie können einen Filter auch verwenden, um inkrementelle

Validierungen durchzuführen, indem Sie Partitionsinformationen auswählen und diese als Parameter über API-Aufrufe übergeben. Um die Leistung zu verbessern, können Sie ein Partitionsprädikat bereitstellen.

10. Wählen Sie Ausführen aus. Sie sollten Ihre neue Aufgabe in der Liste Data quality task runs (Ausführungen der Datenqualitätsaufgabe) sehen. Wenn in der Spalte Ausführungsstatus für die Aufgabe Abgeschlossen angezeigt wird, können Sie die Ergebnisse des Qualitätsfaktors anzeigen. Möglicherweise müssen Sie Ihr Konsolenfenster aktualisieren, damit der Status korrekt aktualisiert wird.
11. Um die Spalte mit den Details zu den Datenqualitätsergebnissen anzuzeigen, wählen Sie das „+“-Symbol aus, um den Regelsatz zu erweitern. Die Ergebnisse zeigen Ihnen, welche Regeln bei der Auswertung bestanden und fehlgeschlagen sind und was den Regelfehler ausgelöst hat.

Aufrufen des Datenqualitätswerts und der Ergebnisse

So zeigen Sie die neueste Ausführung für alle erstellten Regelsätze an

1. Wählen Sie in der AWS Glue-Konsole im Navigationsbereich Tabellen aus. Wählen Sie dann die Tabelle aus, für die Sie eine Datenqualitätsaufgabe ausführen möchten.
2. Wählen Sie die Registerkarte Datenqualität.
3. Der Snapshot zur Datenqualität zeigt einen allgemeinen Trend der Ausführungen im Laufe der Zeit. Die letzten 10 Ausführungen für alle Regelsätze werden standardmäßig angezeigt. Um nach Regelsatz zu filtern, wählen Sie den gewünschten Regelsatz aus der Dropdown-Liste aus. Bei weniger als 10 Ausführungen werden alle verfügbaren abgeschlossenen Ausführungen angezeigt.
4. In der Tabelle Datenqualität wird jeder Regelsatz mit seiner letzten Ausführung (falls es eine gibt) zusammen mit dem Wert angezeigt. Beim Erweitern des Regelsatzes werden die darin enthaltenen Regeln sowie die Ergebnisse dieser Ausführung angezeigt.

So zeigen Sie die neueste Ausführung eines bestimmten Regelsatzes an

1. Wählen Sie in der AWS Glue-Konsole im Navigationsbereich Tabellen aus. Wählen Sie dann die Tabelle aus, für die Sie eine Datenqualitätsaufgabe ausführen möchten.
2. Wählen Sie die Registerkarte Datenqualität.
3. Wählen Sie in der Tabelle Datenqualität einen bestimmten Regelsatz aus.

4. Wählen Sie auf der Seite mit den Regelsatzdetails die Registerkarte Ausführungsverlauf aus.

   In der Tabelle auf dieser Registerkarte sind alle Ausführungen der Auswertung für diesen bestimmten Regelsatz aufgeführt. Sie können den Verlauf der Wertungen und den Status der Ausführungen anzeigen.
5. Um weitere Informationen zu einer bestimmten Ausführung anzuzeigen, wählen Sie die Ausführungs-ID, um die Seite mit den Details zur Auswertung aufzurufen. Auf dieser Seite können Sie Einzelheiten zur Ausführung und weitere Details über den Status der einzelnen Regelergebnisse anzeigen.

Verwandte Themen

- DQDL-Regeltypreferenz
- Referenz zu Data Quality Definition Language (DQDL)

Bewertung der Datenqualität mit AWS Glue Studio

AWS Glue Data Quality bewertet und überwacht die Qualität Ihrer Daten anhand von Regeln, die Sie definieren. Dies macht es einfach, die Daten zu identifizieren, die Maßnahmen erfordern. In AWS Glue Studio können Sie Datenqualitätsknoten zu Ihrem visuellen Auftrag hinzufügen, um Datenqualitätsregeln für Tabellen in Ihrem Datenkatalog zu erstellen. Anschließend können Sie Änderungen an Ihren Datensätzen im Laufe der Zeit überwachen und bewerten. Einen Überblick über die Arbeit mit AWS Glue Data Quality in AWS Glue Studio finden Sie im folgenden Video.

Im Folgenden sind die allgemeinen Schritte für die Arbeit mit AWS Glue Data Quality aufgeführt:

1. Create data quality rules (Datenqualitätsregeln erstellen) – Erstellen Sie mit dem DQDL-Generator mehrere Datenqualitätsregeln, indem Sie integrierte Regelsätze auswählen, die Sie konfigurieren.
2. Configure a data quality job (Konfigurieren eines Auftrags zur Datenqualität) – Definieren Sie Aktionen auf der Grundlage der Datenqualitätsergebnisse und Ausgabeoptionen.
3. Speichern und Ausführen eines Auftrags mit Datenqualität – Erstellen und führen Sie einen Auftrag aus. Durch das Speichern des Auftrags werden die Regelsätze, die Sie für den Auftrag erstellt haben, gespeichert.
4. Monitor and review the data quality results (Überwachen und Überprüfen der Datenqualitätsergebnisse) – Überprüfen Sie die Datenqualitätsergebnisse nach Abschluss der Auftragsausführung. Optional können Sie den Auftrag für ein zukünftiges Datum planen.

Vorteile

Datenanalysten, Dateningenieure und Datenwissenschaftler können den Knoten Bewertung der Datenqualität in AWS Glue Studio verwenden, um die Qualität von Daten im visuellen Auftrags-Editor zu analysieren, zu konfigurieren, zu überwachen und zu verbessern. Die Verwendung des Knotens für die Datenqualität hat unter anderem folgende Vorteile:

- Sie können Datenqualitätsprobleme erkennen – Sie können nach Problemen suchen, indem Sie Regeln erstellen, die die Merkmale Ihrer Datensätze überprüfen.
- Einfacher Einstieg – Sie können mit vorgefertigten Regeln und Aktionen beginnen.
- Enge Integration – Sie können Datenqualitätsknoten in AWS Glue Studio verwenden, da AWS Glue Data Quality auf dem AWS Glue Data Catalog ausgeführt wird.

Auswertung der Datenqualität für ETL-Aufträge in AWS Glue Studio

In diesem Tutorial beginnen Sie mit AWS Glue Data Quality in AWS Glue Studio. Sie lernen, wie Sie:

- Erstellen Sie Regeln mit dem Regelgenerator der Data Quality Definition Language (DQDL).
- Datenqualitätsaktionen, auszugebende Daten und den Ausgabeort der Datenqualitätsergebnisse festlegen.
- Ergebnisse der Datenqualität überprüfen.

Um anhand eines Beispiels zu üben, lesen Sie den Blogbeitrag Erste Schritte mit AWS Glue Data Quality for ETL-Pipelines.

Schritt 1: Hinzufügen des Transformationsknotens Auswertung von Bewertung der Datenqualität zum visuellen Auftrag

In diesem Schritt fügen Sie dem visuellen Auftrag-Editor den Knoten Bewertung der Datenqualität hinzu.

So fügen Sie den Datenqualitätsknoten hinzu

1. Wählen Sie in der AWS Glue Studio-Konsole im Bereich Job erstellen die Option Visual mit einer Quelle und einem Ziel aus und wählen Sie dann Create.

2. Wählen Sie einen Knoten auf, auf den Sie die Transformation der Datenqualität anwenden möchten. In der Regel handelt es sich dabei um einen Transformationsknoten oder eine Datenquelle.
3. Öffnen Sie das Bedienfeld Ressourcen auf der linken Seite, indem Sie das „+“-Symbol auswählen. Suchen Sie anschließend in der Suchleiste nach Bewertung der Datenqualität und wählen Sie aus den Suchergebnissen Bewertung der Datenqualität aus.
4. Der visuelle Auftrag-Editor zeigt den Transformationsknoten Auswertung der Datenqualität an, der von dem von Ihnen ausgewählten Knoten abzweigt. Auf der rechten Seite der Konsole wird die Registerkarte Transform (Transformieren) automatisch geöffnet. Wenn Sie den übergeordneten Knoten ändern müssen, wählen Sie die Registerkarte Knoteneigenschaften und dann den übergeordneten Knoten aus dem Dropdown-Menü aus.

   Wenn Sie einen neuen übergeordneten Knoten auswählen, wird eine neue Verbindung zwischen dem übergeordneten Knoten und dem Knoten Evaluate Data Quality (Bewertung der Datenqualität) hergestellt. Entfernen Sie alle unerwünschten übergeordneten Knoten. Es kann nur ein übergeordneter Knoten mit einem Knoten Evaluate Data Quality (Bewertung der Datenqualität) verbunden sein.
5. Die Transformation Bewertung der Datenqualität unterstützt mehrere übergeordnete Ebenen, so dass Sie die Datenqualitätsregeln für mehrere Datensätze auswerten können. Zu den Regeln, die mehrere Datensätze unterstützen ReferentialIntegrity, gehören DatasetMatch, SchemaMatch RowCountMatch, und AggregateMatch.

   Wenn Sie der Transformation Bewertung der Datenqualität mehrere Eingaben hinzufügen, müssen Sie Ihre „primäre“ Eingabe auswählen. Ihre primäre Eingabe ist der Datensatz, dessen Datenqualität Sie validieren möchten. Alle anderen Knoten oder Eingaben werden als Referenzen behandelt.

   Sie können die Transformation Bewertung der Datenqualität verwenden, um bestimmte Datensätze zu identifizieren, die die Datenqualitätsprüfungen nicht bestanden haben. Wir empfehlen Ihnen, Ihren primären Datensatz auszuwählen, da dem primären Datensatz neue Spalten hinzugefügt werden, die fehlerhafte Datensätze kennzeichnen.
6. Sie können Aliase für Eingabedatenquellen angeben. Aliase bieten eine weitere Möglichkeit, auf die Eingabequelle zu verweisen, wenn Sie die ReferentialIntegrity Regel verwenden. Da nur eine Datenquelle als primäre Quelle festgelegt werden kann, ist für jede weitere Datenquelle, die Sie hinzufügen, ein Alias erforderlich.

Im folgenden Beispiel gibt die ReferentialIntegrity Regel die Eingabedatenquelle anhand des Aliasnamens an und führt einen one-to-one Vergleich mit der primären Datenquelle durch.

```
Rules = [
 ReferentialIntegrity “Aliasname.name” = 1
]
```

Schritt 2: Erstellen einer Regel mit DQDL

In diesem Schritt erstellen Sie eine Regel mit DQDL. Für dieses Tutorial erstellen Sie eine einzelne Regel mit dem Regeltyp Vollständigkeit. Dieser Regeltyp prüft den Prozentsatz vollständiger (nicht Null-)Werte in einer Spalte anhand eines bestimmten Ausdrucks. Weitere Informationen zur Verwendung von DQDL finden Sie unter DQDL.

1. Fügen Sie auf der Registerkarte Transformieren einen Regeltyp hinzu, indem Sie auf die Schaltfläche Einfügen klicken. Dadurch wird der Regeltyp zum Regeleditor hinzugefügt, wo Sie die Parameter für die Regel eingeben können.

> **Note**
>
> Stellen Sie beim Bearbeiten von Regeln sicher, dass die Regeln in Klammern stehen und durch Kommas getrennt sind. Ein vollständiger Regelausdruck sieht beispielsweise wie folgt aus:
>
> ```
> Rules= [
>     Completeness "year">0.8, Completeness "month">0.8
> ]
> ```
>
> In diesem Beispiel wird der Parameter für die Vollständigkeit der Spalten „Jahr“ und „Monat“ angegeben. Damit die Regel gilt, müssen diese Spalten zu mehr als 80 % „vollständig“ sein oder in über 80 % der Fälle Daten für die jeweilige Spalte enthalten.

Suchen Sie in diesem Beispiel nach dem Regeltyp Completeness (Vollständigkeit) und fügen Sie ihn ein. Dadurch wird der Regeltyp zum Regeleditor hinzugefügt. Dieser Regeltyp verfügt über die folgende Syntax: `Completeness <COL_NAME> <EXPRESSION>`.

Bei den meisten Regeltypen ist es erforderlich, dass Sie einen Ausdruck als Parameter angeben, um eine boolesche Antwort zu erstellen. Weitere Informationen zu unterstützten DQDL-Ausdrücken finden Sie unter DQDL-Ausdrücke. Als Nächstes fügen Sie den Spaltennamen hinzu.

2. Wählen Sie im DQDL-Regelgenerator die Registerkarte Schema aus. Verwenden Sie die Suchleiste, um den Spaltennamen im Eingabeschema zu finden. Das Eingabeschema zeigt den Spaltennamen und den Datentyp an.
3. Klicken Sie im Regeleditor rechts neben den Regeltyp, um den Cursor dort einzufügen, wo die Spalte eingefügt werden soll. Alternativ können Sie den Namen der Spalte in der Regel eingeben.

   Wählen Sie beispielsweise aus der Liste der Spalten in der Eingabeschemaliste die Schaltfläche Einfügen neben der Spalte (in diesem Beispiel Jahr) aus. Dadurch wird die Spalte zur Regel hinzugefügt.
4. Fügen Sie dann im Regeleditor einen Ausdruck hinzu, um die Regel auszuwerten. Da der Regeltyp Vollständigkeit den Prozentsatz vollständiger (nicht Null) Werte in einer Spalte anhand eines bestimmten Ausdrucks prüft, geben Sie einen Ausdruck wie z. B. `> 0.8` ein. Diese Regel überprüft die Spalte, wenn sie zu mehr als 80 % vollständige Werte (nicht Null) enthält.

Schritt 3: Konfigurieren von Ausgaben für Datenqualität

Nach dem Erstellen von Datenqualitätsregeln können Sie zusätzliche Optionen auswählen, um die Ausgabe des Datenqualitätsknotens festzulegen.

1. Wählen Sie unter Ausgabe der Datenqualitätstransformation eine der folgenden Optionen aus:

   - Originaldaten – Wählen Sie diese Option, um Originaleingabedaten auszugeben. Wenn Sie diese Option wählen, wird dem Auftrag ein neuer untergeordneter Knoten „rowLevelOutcomes“ hinzugefügt. Das Schema stimmt mit dem Schema des primären Datensatzes überein, der als Eingabe an die Transformation übergeben wurde. Diese Option ist nützlich, wenn Sie nur die Daten weiterleiten möchten und der Auftrag bei Qualitätsproblemen fehlschlägt.

Ein weiterer Anwendungsfall besteht darin, fehlerhafte Datensätze zu erkennen, die die Datenqualitätsprüfungen nicht bestanden haben. Um fehlerhafte Datensätze zu erkennen, wählen Sie die Option Neue Spalten hinzufügen, um Datenqualitätsfehler anzuzeigen. Diese Aktion fügt dem Schema der Transformation „rowLevelOutcomes“ vier neue Spalten hinzu.

- DataQualityRulesPass(String-Array) — Stellt eine Reihe von Regeln bereit, die die Datenqualitätsprüfungen bestanden haben.
- DataQualityRulesFail(String-Array) — Stellt eine Reihe von Regeln bereit, die Datenqualitätsprüfungen nicht bestanden haben.
- DataQualityRulesSkip(String-Array) — Stellt eine Reihe von Regeln bereit, die übersprungen wurden. Die folgenden Regeln können Fehlerdatensätze nicht identifizieren, da sie auf Datensatzebene angewendet werden.
  - AggregateMatch
  - ColumnCount
  - ColumnExists
  - ColumnNamesMatchPattern
  - CustomSql
  - RowCount
  - RowCountMatch
  - StandardDeviation
  - Mean
  - ColumnCorrelation
- DataQualityEvaluationResult— Stellt den Status „Bestanden“ oder „Fehlgeschlagen“ auf Zeilenebene bereit. Beachten Sie, dass Ihre Gesamtergebnisse den Wert FEHLGESCHLAGEN haben können, ein bestimmter Datensatz jedoch möglicherweise bestanden hat. Beispielsweise könnte die RowCount Regel fehlgeschlagen sein, aber alle anderen Regeln könnten erfolgreich gewesen sein. In solchen Fällen lautet der Feldstatus „Bestanden“.

2. Datenqualitätsergebnisse – Wählen Sie, ob Sie die konfigurierten Regeln und deren Status als bestanden oder nicht bestanden ausgeben möchten. Diese Option ist nützlich, wenn Sie Ihre Ergebnisse in Amazon S3 oder andere Datenbanken schreiben möchten.
3. Einstellungen für die Datenqualitätsausgabe (Optional) – Wählen Sie Einstellungen für die Datenqualitätsausgabe um das Feld Speicherort der Datenqualitätsergebnisse anzuzeigen.

Wählen Sie dann Durchsuchen aus, um nach einem Amazon-S3-Speicherort zu suchen, den Sie als Datenqualitätsausgabeziel festlegen möchten.

Schritt 4. Datenqualitätsaktionen konfigurieren

Sie können Aktionen verwenden, um Metriken zu veröffentlichen CloudWatch oder Jobs auf der Grundlage bestimmter Kriterien zu beenden. Aktionen sind erst verfügbar, nachdem Sie eine Regel erstellt haben. Wenn Sie diese Option wählen, werden dieselben Metriken auch in Amazon EventBridge veröffentlicht. Mit diesen Optionen können Sie Warnmeldungen zur Benachrichtigung erstellen.

- Bei Ausfall eines Regelsatzes – Sie können auswählen, was geschehen soll, wenn ein Regelsatz während der Ausführung des Auftrags fehlschlägt. Wenn Sie möchten, dass der Auftrag fehlschlägt, wenn die Datenqualität fehlschlägt, legen Sie fest, wann der Auftrag fehlschlagen soll, indem Sie eine der folgenden Optionen auswählen. Standardmäßig ist diese Aktion nicht ausgewählt und der Auftrag schließt seine Ausführung auch dann ab, wenn die Datenqualitätsregeln fehlschlagen.
  - Keine – Wenn Sie Keine (Standardeinstellung) auswählen, schlägt der Auftrag nicht fehl und wird trotz Regelsatzfehlern weiter ausgeführt.
  - Auftrag nach dem Laden von Daten auf das Ziel fehlschlagen – Der Auftrag schlägt fehl und es werden keine Daten gespeichert. Um die Ergebnisse zu speichern, wählen Sie einen Amazon-S3-Speicherort aus, an dem die Datenqualitätsergebnisse gespeichert werden.
  - Auftrag fehlschlagen, ohne in die Zieldaten zu laden – Diese Option lässt den Auftrag sofort fehlschlagen, wenn ein Datenqualitätsfehler auftritt. Es werden keine Datenziele geladen, einschließlich der Ergebnisse der Datenqualitätstransformation.

Schritt 5: Anzeigen von Datenqualitätsergebnissen

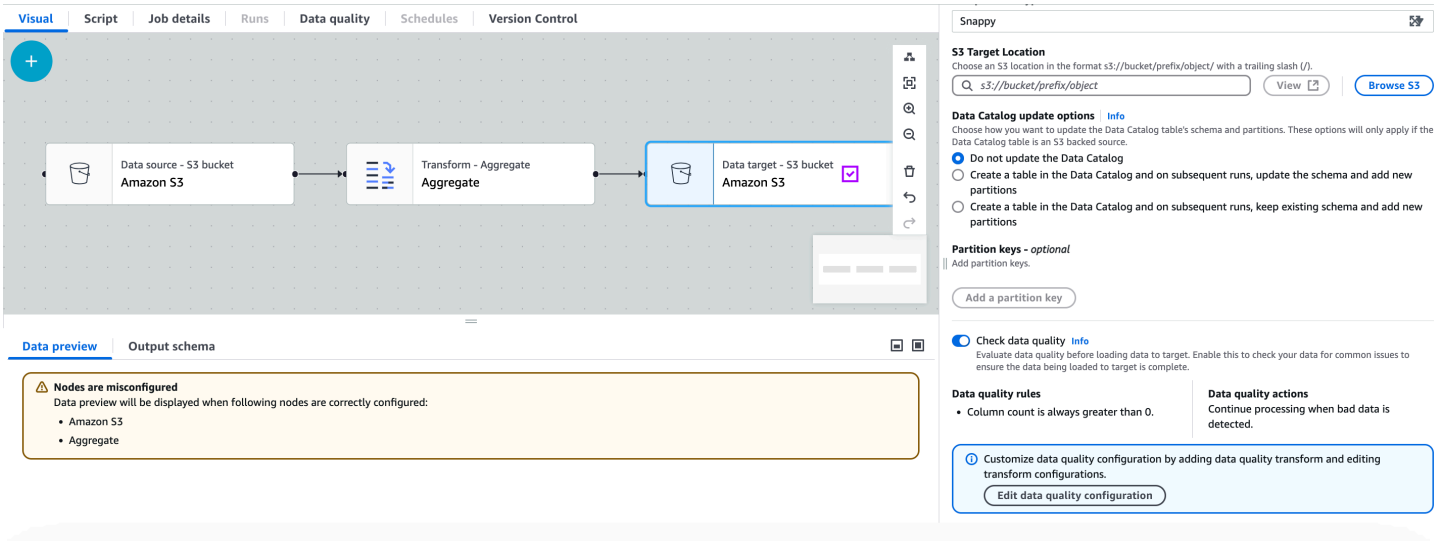
Nachdem Sie den Auftrag ausgeführt haben, können Sie die Datenqualitätsergebnisse anzeigen, indem Sie die Registerkarte Datenqualität auswählen.

1. Zeigen Sie für jede Auftragsausführung die Datenqualitätsergebnisse an. Jeder Knoten zeigt einen Datenqualitätsstatus und Statusdetails an. Wählen Sie einen Knoten aus, um alle Regeln und den Status jeder Regel anzuzeigen.
2. Klicken Sie auf Ergebnisse herunterladen, um eine CSV-Datei herunterzuladen, die Informationen über die Auftragsausführung und Datenqualitätsergebnisse enthält.

3. Wenn Sie mehr als einen Auftrag mit Datenqualitätsergebnissen ausführen lassen, können Sie die Ergebnisse nach Datum und Zeitspanne filtern. Wählen Sie Nach Datum und Uhrzeit filtern aus, um das Filterfenster zu erweitern.
4. Wählen Sie einen relativen Bereich oder einen absoluten Bereich aus. Verwenden Sie für absolute Bereiche den Kalender, um ein Datum auszuwählen und Werte für Startzeit und Endzeit einzugeben. Wenn Sie fertig sind, wählen Sie Anwenden.

Automatische Datenqualität

Wenn Sie einen AWS Glue-ETL-Job mit Amazon S3 als Ziel erstellen, aktiviert AWS Glue ETL automatisch eine Datenqualitätsregel, die prüft, ob die geladenen Daten mindestens eine Spalte haben. Diese Regel soll sicherstellen, dass die geladenen Daten nicht leer oder beschädigt sind. Wenn diese Regel jedoch fehlschlägt, schlägt der Job nicht fehl. Stattdessen werden Sie eine Verringerung Ihres Datenqualitätsfaktors feststellen. Darüber hinaus ist die Anomalieerkennung standardmäßig aktiviert, wodurch die Anzahl der Spalten in den Daten überwacht wird. Wenn es Abweichungen oder Abnormalitäten bei der Anzahl der Spalten gibt, wird AWS Glue ETL Sie über diese Anomalien informieren. Diese Funktion hilft Ihnen, potenzielle Probleme mit den Daten zu identifizieren und geeignete Maßnahmen zu ergreifen. Um die Datenqualitätsregel und ihre Konfiguration anzuzeigen, können Sie in Ihrem AWS Glue ETL-Job auf das Amazon S3 S3-Ziel klicken. Die Konfiguration der Regel wird angezeigt, wie im bereitgestellten Screenshot gezeigt.



Sie können zusätzliche Datenqualitätsregeln hinzufügen, indem Sie Datenqualitätskonfiguration bearbeiten auswählen.

Aggregierte Metriken

Für die Erstellung von Dashboards benötigen Sie möglicherweise aggregierte Metriken, z. B. die Anzahl der Datensätze, die bestanden, fehlgeschlagen oder übersprungen wurden, auf Regel- oder Regelsatzebene. Um die aggregierten Metriken und Regelmetriken für jede Regel abzurufen, aktivieren Sie zunächst Aggregierte Metriken, indem Sie die Option zu Ihrer Funktion hinzufügen. `publishAggregatedMetrics` `EvaluateDataQuality`

Die möglichen Optionen für `additional_options` `publishAggregatedMetrics` sind `ENABLED` und. `DISABLED` Beispiel:

```
EvaluateDataQualityMultiframe = EvaluateDataQuality().process_rows(
    frame=medicare_dyf,
    ruleset=EvaluateDataQuality_ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",
        "enableDataQualityCloudWatchMetrics": False,
        "enableDataQualityResultsPublishing": False,
    },
    additional_options={"publishAggregatedMetrics.status": "ENABLED"},
)
```

Falls nicht angegeben, `publishAggregatedMetrics.status` ist dies die `DISABLED` Standardeinstellung und die RuleMetrics und die aggregierten Metriken würden jetzt berechnet. Diese Funktion wird derzeit in AWS Glue Interactive Sessions und in Glue ETL-Jobs unterstützt. Dies wird in Glue Catalog Data Quality nicht unterstützt APIs.

Abrufen aggregierter Metrikergebnisse

Wann `additionalOptions` `"publishAggregatedMetrics.status": "ENABLED"` können Sie die Ergebnisse an zwei Stellen abrufen:

1. `AggregatedMetrics`und `RuleMetrics` werden `GetDataQualityResult()` bei Angabe des `resultId` Where zurückgegeben `AggregatedMetrics` und `RuleMetrics` beinhalten:

   Aggregierte Metriken:

   - Gesamtzahl der verarbeiteten Zeilen
   - Gesamtzahl der übergebenen Zeilen
   - Gesamtzahl der fehlgeschlagenen Zeilen
   - Gesamtzahl der verarbeiteten Regeln

- Gesamtzahl der verabschiedeten Regeln
- Gesamtzahl der fehlgeschlagenen Regeln

```
[redacted] % aws --region us-east-1 glue get-data-quality-result --result-id "dqresult-09655f0234b3e3d129381bd162d0347ed25d1c05"
{
    "ResultId": "dqresult-09655f0234b3e3d129381bd162d0347ed25d1c05",
    "ProfileId": "dqprofile-[redacted]",
    "Score": 0.6666666666666666,
    "RulesetName": "EvaluateDataQuality_node1741974822533",
    "EvaluationContext": "EvaluateDataQuality_node1741974822533",
    "StartedOn": 1750687515.374,
    "CompletedOn": 1750687537.485,
    "JobName": "[redacted]_aggregatedmetrics_test",
    "JobRunId": "jr_[redacted]",
    "RuleResults": [
        {
            "Name": "Rule_1",
            "Description": "IsUnique \"customer_identifier\"",
            "Result": "PASS",
            "EvaluatedMetrics": {
                "Column.customer_identifier.Uniqueness": 1.0
            },
            "EvaluatedRule": "IsUnique \"customer_identifier\"",
            "RuleMetrics": {
                "rows_passed": 10.0,
                "rows_failed": 0.0,
                "rows_skipped": 0.0,
                "total_rows": 10.0
            }
        },
        {
            "Name": "Rule_2",
            "Description": "RowCount > 10",
            "EvaluationMessage": "Value: 10 does not meet the constraint requirement!",
            "Result": "FAIL",
            "EvaluatedMetrics": {
                "Dataset.*.RowCount": 10.0
            },
            "EvaluatedRule": "RowCount > 10",
            "RuleMetrics": {
                "rows_passed": 0.0,
                "rows_failed": 0.0,
                "rows_skipped": 10.0,
                "total_rows": 10.0
            }
        },
        {
            "Name": "Rule_3",
            "Description": "Completeness \"customer_identifier\" > 0.5",
            "Result": "PASS",
            "EvaluatedMetrics": {
                "Column.customer_identifier.Completeness": 1.0
            },
            "EvaluatedRule": "Completeness \"customer_identifier\" > 0.5",
            "RuleMetrics": {
                "rows_passed": 10.0,
                "rows_failed": 0.0,
                "rows_skipped": 0.0,
                "total_rows": 10.0
            }
        }
    ],
    "AggregatedMetrics": {
        "TotalRowsProcessed": 10.0,
        "TotalRowsPassed": 10.0,
        "TotalRowsFailed": 0.0,
        "TotalRulesProcessed": 3.0,
        "TotalRulesPassed": 2.0,
        "TotalRulesFailed": 1.0
    }
}
```

Auf Regelebene werden außerdem die folgenden Metriken bereitgestellt:

Regelmetriken:

- Vergangene Zeilen
- Zeilen sind fehlgeschlagen
- Zeile übersprungen

- Gesamtzahl der verarbeiteten Zeilen

2. `AggregatedMetrics`wird als zusätzlicher Datenrahmen zurückgegeben und der `RuleOutcomes` Datenrahmen wird um folgende `RuleMetrics` Daten erweitert:

Beispiel für eine Implementierung

Das folgende Beispiel zeigt, wie aggregierte Metriken in Scala implementiert werden:

```
// Script generated for node Evaluate Data Quality
val EvaluateDataQuality_node1741974822533_ruleset = """
  # Example rules: Completeness "colA" between 0.4 and 0.8, ColumnCount > 10
  Rules = [
      IsUnique "customer_identifier",
      RowCount > 10,
      Completeness "customer_identifier" > 0.5
  ]
"""

val EvaluateDataQuality_node1741974822533 =
 EvaluateDataQuality.processRows(frame=ChangeSchema_node1742850392012,
 ruleset=EvaluateDataQuality_node1741974822533_ruleset,
 publishingOptions=JsonOptions("""{"dataQualityEvaluationContext":
 "EvaluateDataQuality_node1741974822533", "enableDataQualityCloudWatchMetrics":
 "true", "enableDataQualityResultsPublishing": "true"}"""),
 additionalOptions=JsonOptions("""{"compositeRuleEvaluation.method":"ROW","observations.scope":
 "publishAggregatedMetrics.status": "ENABLED"}""" ))

println("--------------------------------ROW LEVEL
 OUTCOMES--------------------------------")
val rowLevelOutcomes_node = EvaluateDataQuality_node1741974822533("rowLevelOutcomes")

rowLevelOutcomes_node.show(10)

 println("--------------------------------RULE LEVEL
 OUTCOMES--------------------------------")

val ruleOutcomes_node = EvaluateDataQuality_node1741974822533("ruleOutcomes")

ruleOutcomes_node.show()

 println("--------------------------------AGGREGATED
 METRICS--------------------------------")
```

```
val aggregatedMetrics_node = EvaluateDataQuality_node1741974822533("aggregatedMetrics")

aggregatedMetrics_node.show()
```

Beispielergebnisse

Die Ergebnisse werden wie folgt zurückgegeben:


```
{
    "Rule": "IsUnique \"customer_identifier\"",
    "Outcome": "Passed",
    "FailureReason": null,
    "EvaluatedMetrics": {
        "Column.customer_identifier.Uniqueness": 1
    },
    "EvaluatedRule": "IsUnique \"customer_identifier\"",
    "PassedCount": 10,
    "FailedCount": 0,
    "SkippedCount": 0,
    "TotalCount": 10
}
{
    "Rule": "RowCount > 10",
    "Outcome": "Failed",
    "FailureReason": "Value: 10 does not meet the constraint requirement!",
    "EvaluatedMetrics": {
        "Dataset.*.RowCount": 10
    },
    "EvaluatedRule": "RowCount > 10",
    "PassedCount": 0,
    "FailedCount": 0,
    "SkippedCount": 10,
    "TotalCount": 10
}
{
    "Rule": "Completeness \"customer_identifier\" > 0.5",
    "Outcome": "Passed",
    "FailureReason": null,
    "EvaluatedMetrics": {
        "Column.customer_identifier.Completeness": 1
    },
    "EvaluatedRule": "Completeness \"customer_identifier\" > 0.5",
    "PassedCount": 10,
```

```
    "FailedCount": 0,
    "SkippedCount": 0,
    "TotalCount": 10
}
```

Die aggregierten Metriken lauten wie folgt:

```
{ "TotalRowsProcessed": 10, "PassedRows": 10, "FailedRows": 0, "TotalRulesProcessed":
 3, "RulesPassed": 2, "RulesFailed": 1 }
```

Regelgenerator für Data Quality

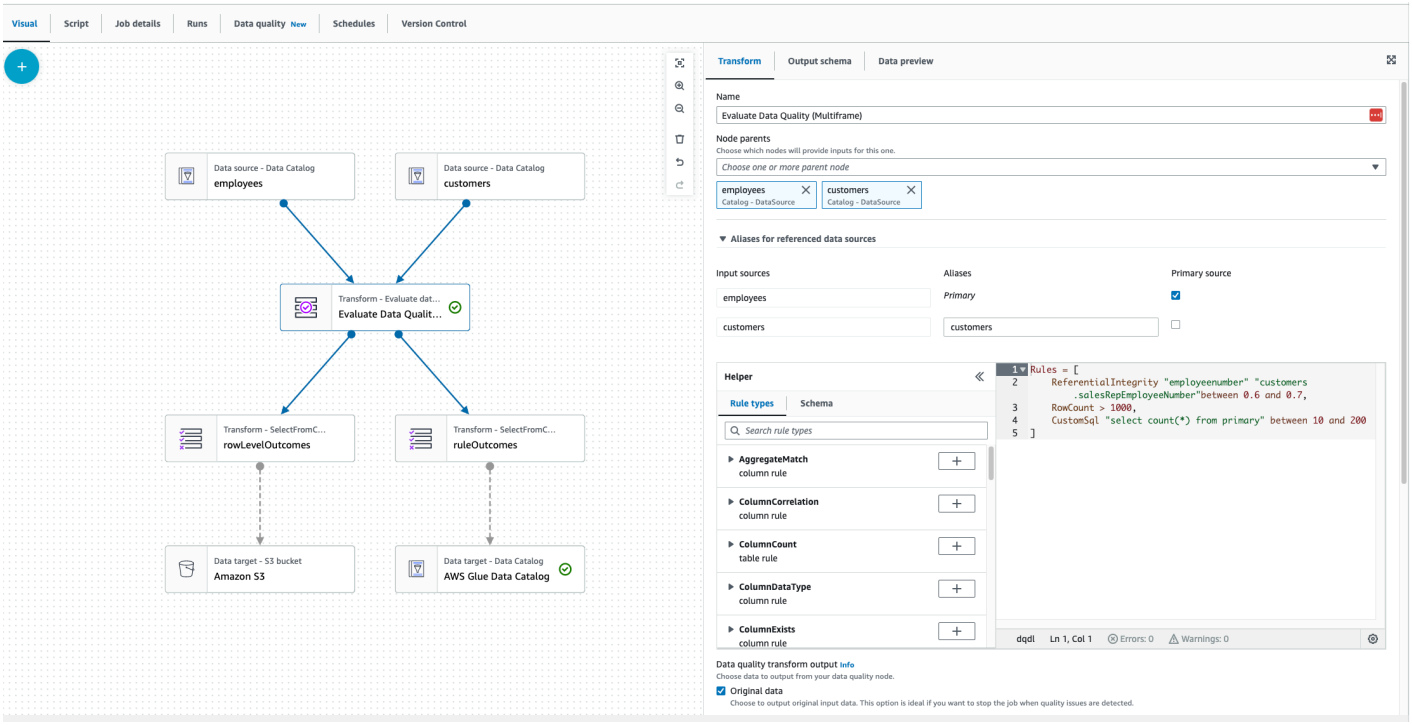
Mit dem Regelgenerator der Data Quality Definition Language (DQDL) können Sie Datenqualitätsregeln zur Auswertung Ihrer Daten erstellen. Wählen Sie zunächst einen Regeltyp aus und geben Sie dann die Parameter im Regeleditor an. Der Regeleditor zeigt Ihnen auch alle Fehler und Warnungen an, während Sie Regeln erstellen.

Das DQDL-Handbuch enthält eine umfassende Dokumentation zum Erstellen von Regeln mithilfe der DQDL-Syntax, integrierten Regeltypen und Beispielen.

Knoten für Bewertung der Datenqualität

Beim Arbeiten mit dem Transformationsknoten Bewertung der Datenqualität und dem DQDL-Regelgenerator können Sie den Arbeitsbereich erweitern.

- Um die Registerkarte Transformieren so zu erweitern, dass sie den gesamten Bildschirm ausfüllt, wählen Sie das Erweiterungssymbol in der oberen rechten Ecke des Bereichs Knotendetails.
- Um den DQDL-Regeleditor zu erweitern, wählen Sie das Symbol **<<**, um den Regeleditor zu erweitern und die Registerkarten Regeltypen und Schema zu reduzieren.



Komponenten

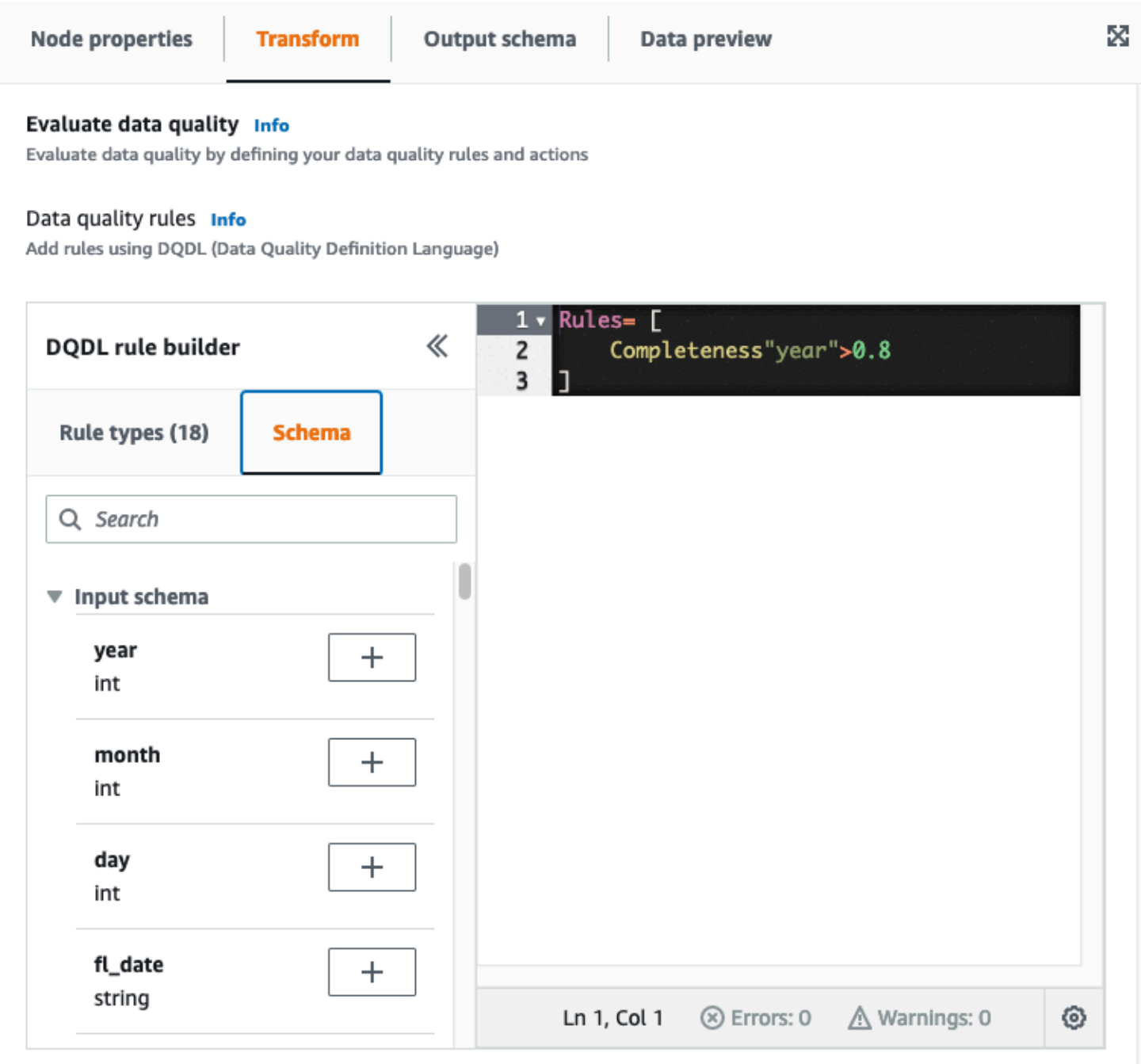
Es gibt 26 Regeltypen, die eingebaut sind AWS Glue Studio. Jeder Regeltyp hat eine Beschreibung und Beispiele, wie er verwendet werden kann.

Regeltypen für die Datenqualität

AWS Glue Studio bietet integrierte Regeltypen, um die Erstellung einer Regel zu vereinfachen. Weitere Informationen zu Regeltypen finden Sie unter Referenz zu DQDL-Regeltypen.

Schema

Auf der Registerkarte Schema werden die Spaltennamen und der Datentyp des übergeordneten Knotens angezeigt. Es werden Schemas mehrerer Knoten angezeigt. Sie können das Eingabeschema anzeigen, nach Spaltennamen suchen und die Spalte in den Regeleditor einfügen.



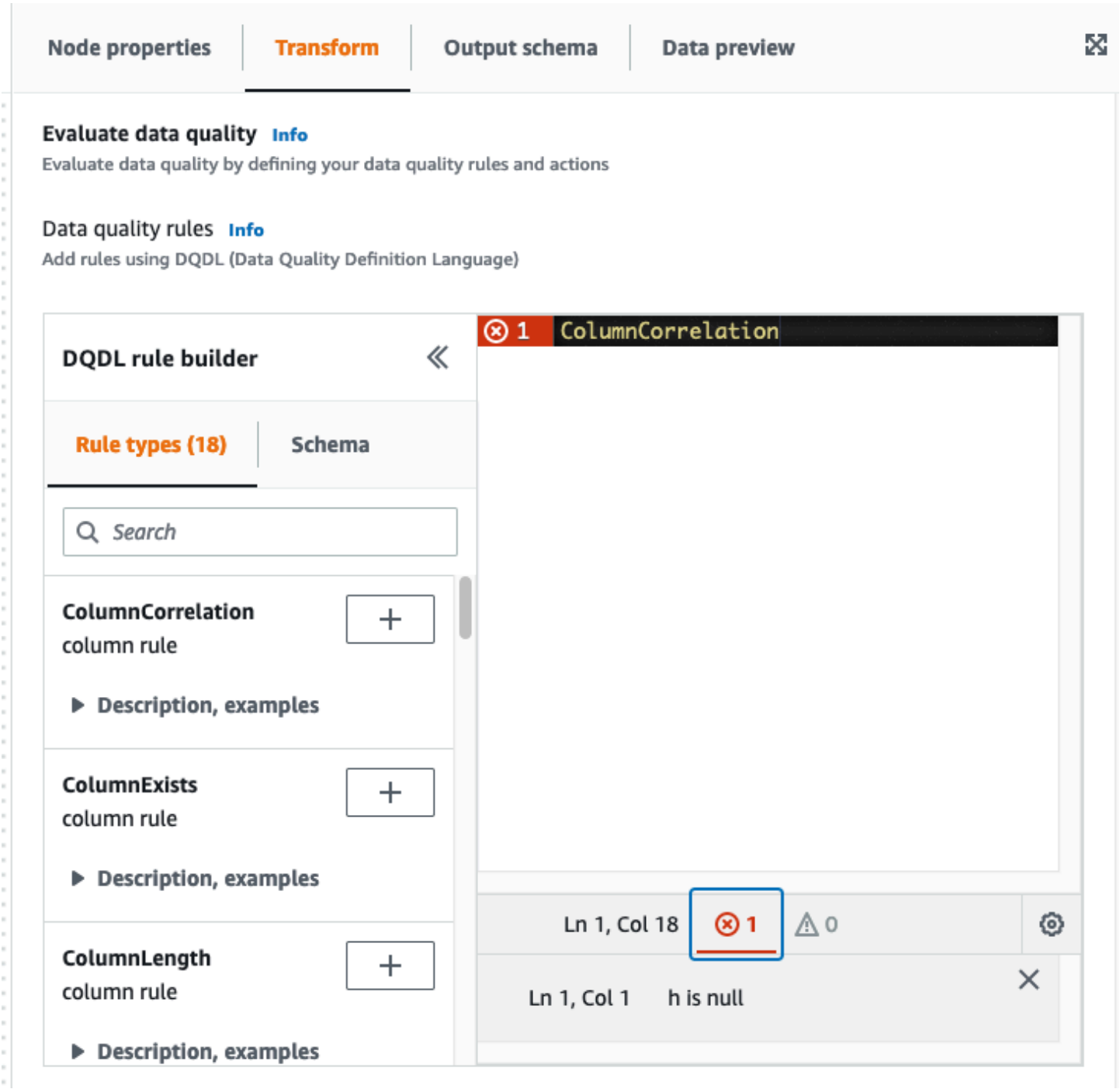
Regeleditor

Der Regeleditor ist ein Texteditor, in dem Sie Regeln schreiben und bearbeiten können. Wenn Sie einen Regeltyp aus dem DQDL Regelgenerator auswählen, wird der Regeltyp dem Regel-Editor hinzugefügt. Anschließend können Sie Parameter angeben, Regeln hinzufügen und Regeln nach Bedarf bearbeiten, indem Sie den Text ändern. AWS Glue Studio validiert die Regeln im Regeleditor und zeigt etwaige Fehler und Warnungen an.

Errors and warnings (Fehler und Warnungen)

Wenn eine Regel nicht der DQDL-Regelsyntax entspricht, zeigt der Regeleditor mehrere visuelle Indikatoren an, die darauf hinweisen, dass ein Fehler vorliegt:

- Der Regeleditor zeigt ein Fehlersymbol und eine rote Farbe in der Zeile mit dem Fehler an.
- Der Regeleditor zeigt die Anzahl der Fehler neben dem roten Fehlersymbol an.
- Wenn Sie die Zeile mit dem Fehler auswählen, werden unten im Regeleditor Beschreibungen des Fehlers und der Fehlerstelle (Zeile und Spalte) angezeigt.



Maßnahmen zur Datenqualität

Standardmäßig ist diese Aktion nicht ausgewählt und der Auftrag wird seine Ausführung auch dann beenden, wenn die Datenqualitätsregeln fehlschlagen.

Wählen Sie zwischen den folgenden Aktionen. Sie können Aktionen verwenden, um Ergebnisse zu veröffentlichen CloudWatch oder Jobs auf der Grundlage bestimmter Kriterien zu beenden. Aktionen sind erst verfügbar, nachdem Sie eine Regel erstellt haben.

- Ergebnisse veröffentlichen in CloudWatch — Wenn Sie einen Job ausführen, fügen Sie die Ergebnisse zu hinzu CloudWatch.
- Auftrag fehlschlagen, wenn die Datenqualität fehlschlägt – Wenn die Datenqualitätsregeln fehlschlagen, wird auch der Auftrag fehlschlagen.

Ausgabe zur Transformation der Datenqualität

- Originaldaten – Wählen Sie diese Option, um Originaleingabedaten auszugeben. Diese Option ist ideal, wenn Sie den Auftrag anhalten möchten, wenn Qualitätsprobleme erkannt werden.
- Datenqualitätsmetriken – Wählen Sie die Ausgabe von konfigurierten Regeln und deren Status als bestanden oder nicht bestanden. Diese Option ist nützlich, wenn Sie eine benutzerdefinierte Aktion durchführen möchten.

Ausgabeeinstellungen für die Datenqualität

Legen Sie den Speicherort der Datenqualitätsergebnisse fest, indem Sie den Amazon-S3-Speicherort als Datenqualitäts-Ausgabeziel angeben.

Konfiguration der Anomalieerkennung in AWS Glue ETL-Jobs

Um mit der Erkennung von Anomalien in AWS Glue Studio zu beginnen, öffnen Sie einen AWS Glue Studio-Job und klicken Sie auf „Evaluate Data Quality Transform“.

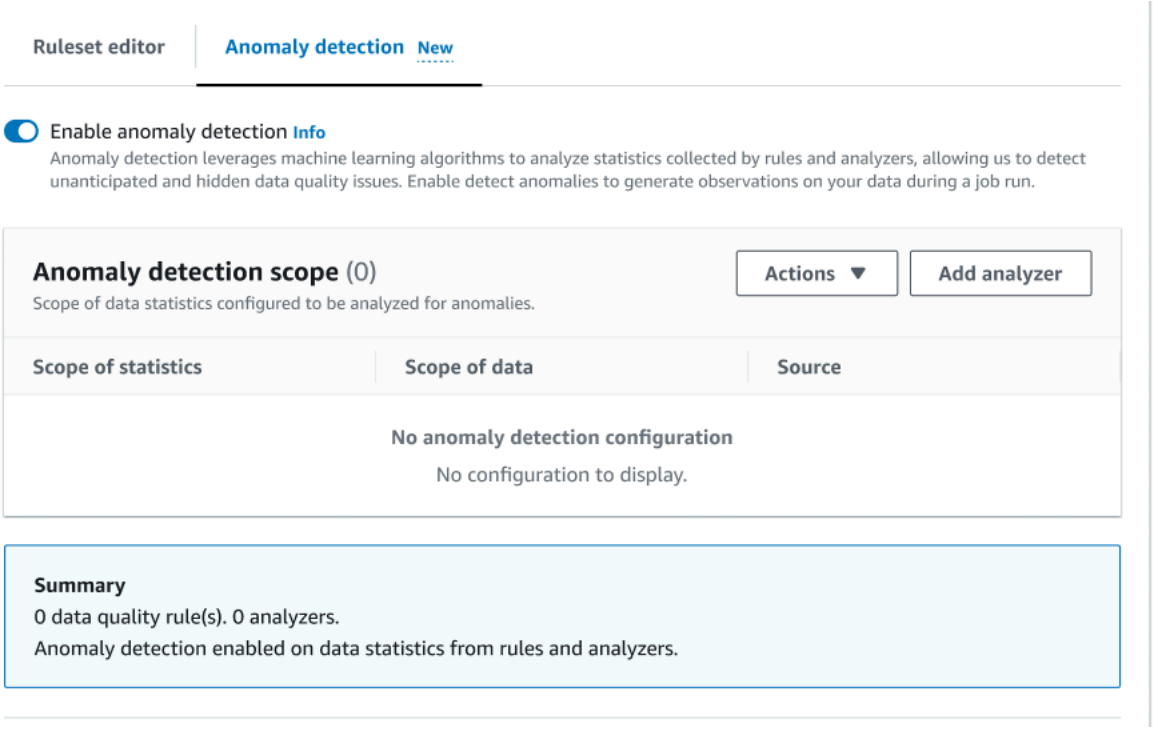
Wenn Sie diese Funktion aktivieren, analysiert AWS Glue Data Quality Ihre Daten im Laufe der Zeit, um Anomalien zu erkennen. Es bietet wertvolle Datenstatistiken und Beobachtungen zu Ihren Daten, sodass Sie bei identifizierten Anomalien Maßnahmen ergreifen können.

Lesen Sie die Dokumentation zur Erkennung von Anomalien, um mehr über die Funktionsweise dieser Funktion zu erfahren.

Aktivierung der Erkennung von Anomalien

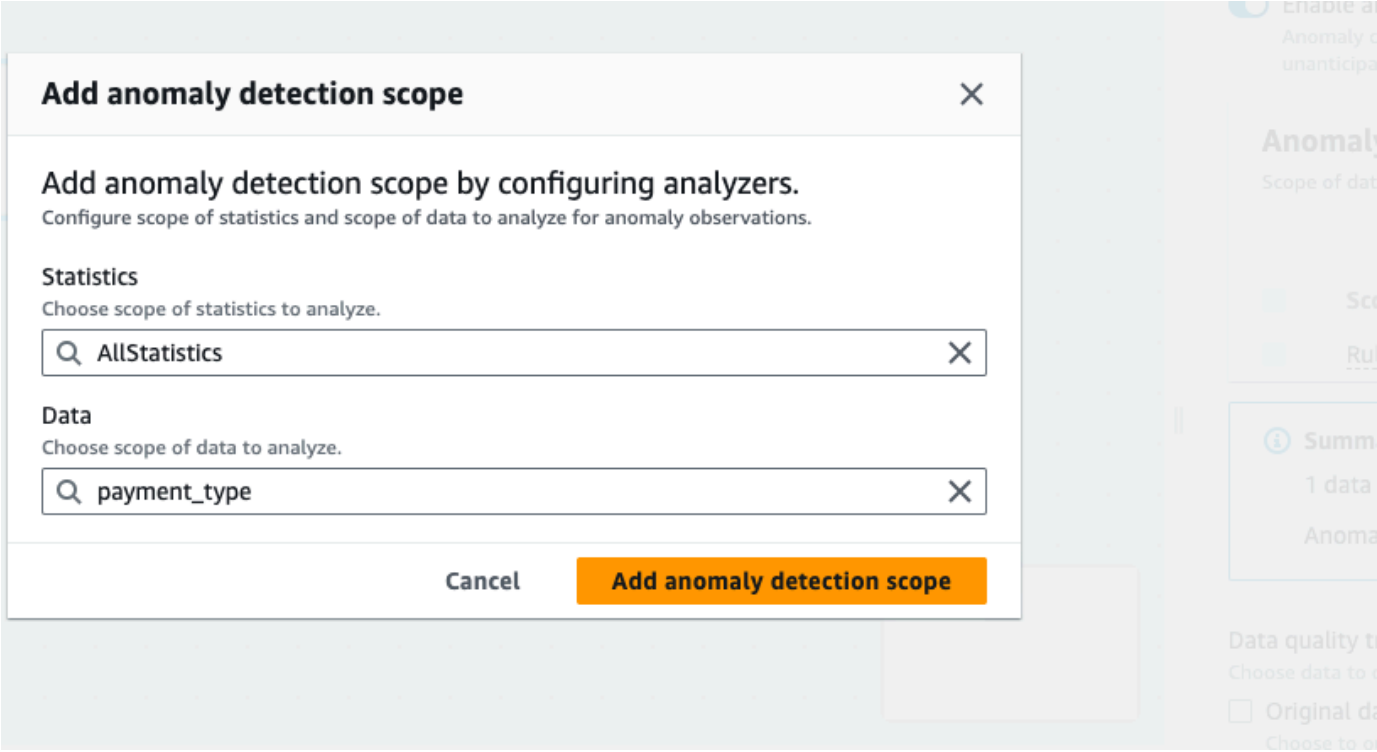
Um die Anomalieerkennung zu aktivieren in AWS Glue Studio:

1. Wählen Sie in Ihrem Auftrag den Knoten Datenqualität und anschließend die Registerkarte Anomalieerkennung aus. Schalten Sie um, um die Option „Anomalieerkennung aktivieren“ zu aktivieren.



2. Legen Sie fest, welche Daten auf Anomalien überwacht werden sollen, indem Sie Analysator hinzufügen auswählen. Es gibt zwei Felder, die Sie ausfüllen können: Statistik und Daten.

   - Statistiken sind Informationen über die Form und andere Eigenschaften Ihrer Daten. Sie können eine oder mehrere Statistiken gleichzeitig auswählen oder Alle Statistiken auswählen. Zu den Statistiken gehören: Vollständigkeit, Eindeutigkeit, Mittelwert, Summe StandardDeviation, Entropie und mehr DistinctValuesCount. UniqueValueRatio Weitere Informationen finden Sie in der Analyzer-Dokumentation.
   - Daten sind die Spalten in Ihrem Datensatz. Sie können alle oder einzelne Spalten auswählen.



3. Wählen Sie Anomalieerkennungsbereich hinzufügen aus, um die Änderungen zu speichern. Nachdem Sie Analysatoren hinzugefügt haben, können Sie sie im Bereich Bereich der Anomalieerkennung sehen.

   Sie können Ihre Analysatoren auch über das Aktionsmenü bearbeiten oder die Registerkarte Regelsatz-Editor auswählen und sie direkt im Notizblock des Regelsatz-Editors bearbeiten. Sie sehen die Analysatoren, die Sie gemäß den von Ihnen erstellten Regeln gespeichert haben.

```
Rules = [

]

Analyzers = [
    Completeness “id”
]
```

Sobald der aktualisierte Regelsatz und die Analysatoren konfiguriert sind, überwacht AWS Glue Data Quality kontinuierlich eingehende Datenströme. Je nach Ihren Einstellungen kann es potenzielle

Anomalien durch Warnmeldungen oder Jobstopps signalisieren. Diese proaktive Überwachung trägt dazu bei, die Datenqualität und -integrität in Ihren Daten-Pipelines sicherzustellen.

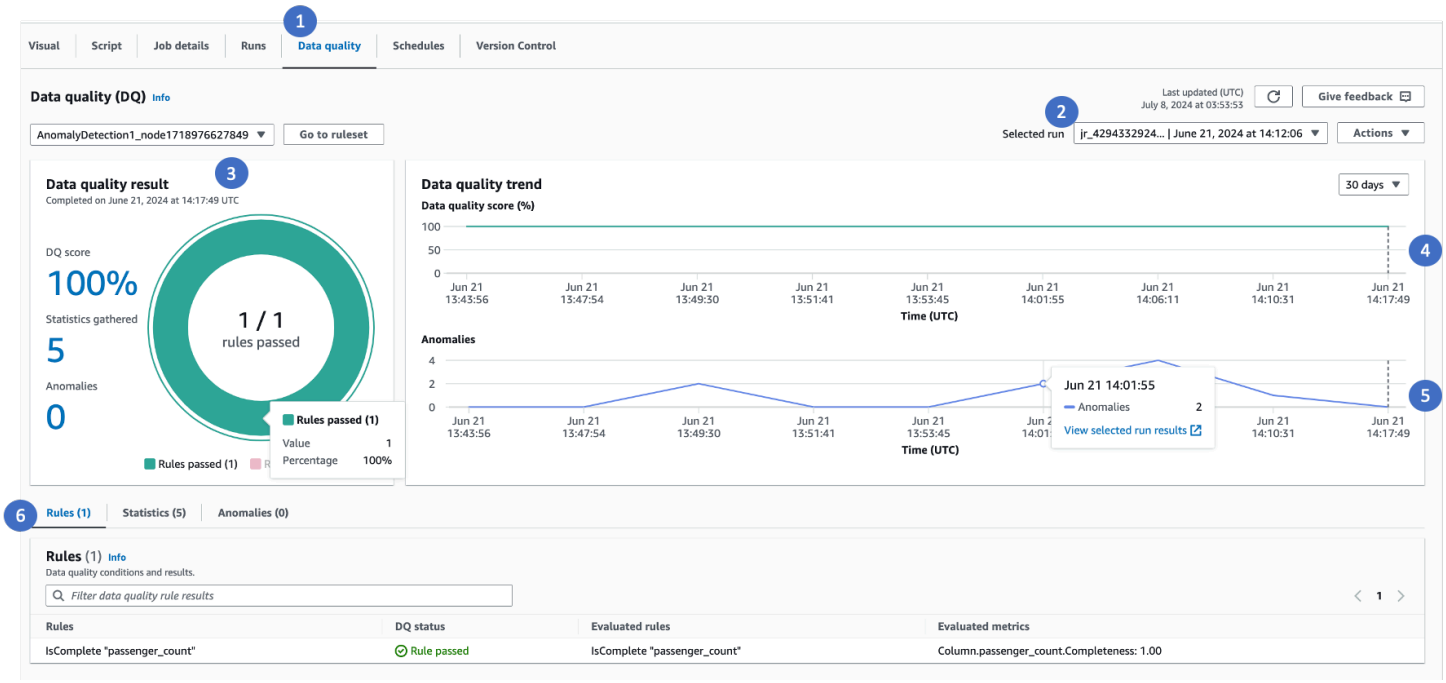
Im nächsten Abschnitt erfahren Sie, wie Sie die vom System identifizierten Anomalien effektiv überwachen können. Sie erfahren auch, wie Sie die von AWS Glue Data Quality gesammelten Datenstatistiken anzeigen und analysieren können. Darüber hinaus erfahren Sie, wie Sie Feedback zu dem Modell des maschinellen Lernens geben können, das die Funktion zur Erkennung von Anomalien unterstützt. Diese Feedback-Schleife ist entscheidend, um die Genauigkeit des Modells zu verbessern und sicherzustellen, dass es Anomalien, die Ihren spezifischen Geschäftsanforderungen und Datenmustern entsprechen, effektiv erkennen kann.

Datenqualitätswerte und Anomalien anzeigen

In diesem Abschnitt werden wir das Datenqualitäts-Dashboard und die verschiedenen Funktionen, die es bietet, untersuchen.

Visualisieren und verstehen Sie wichtige Kennzahlen und Trends zur Datenqualität

Sobald Ihr Job erfolgreich war, wählen Sie die Registerkarte Datenqualität, um die Datenqualitätswerte und Anomalien einzusehen.

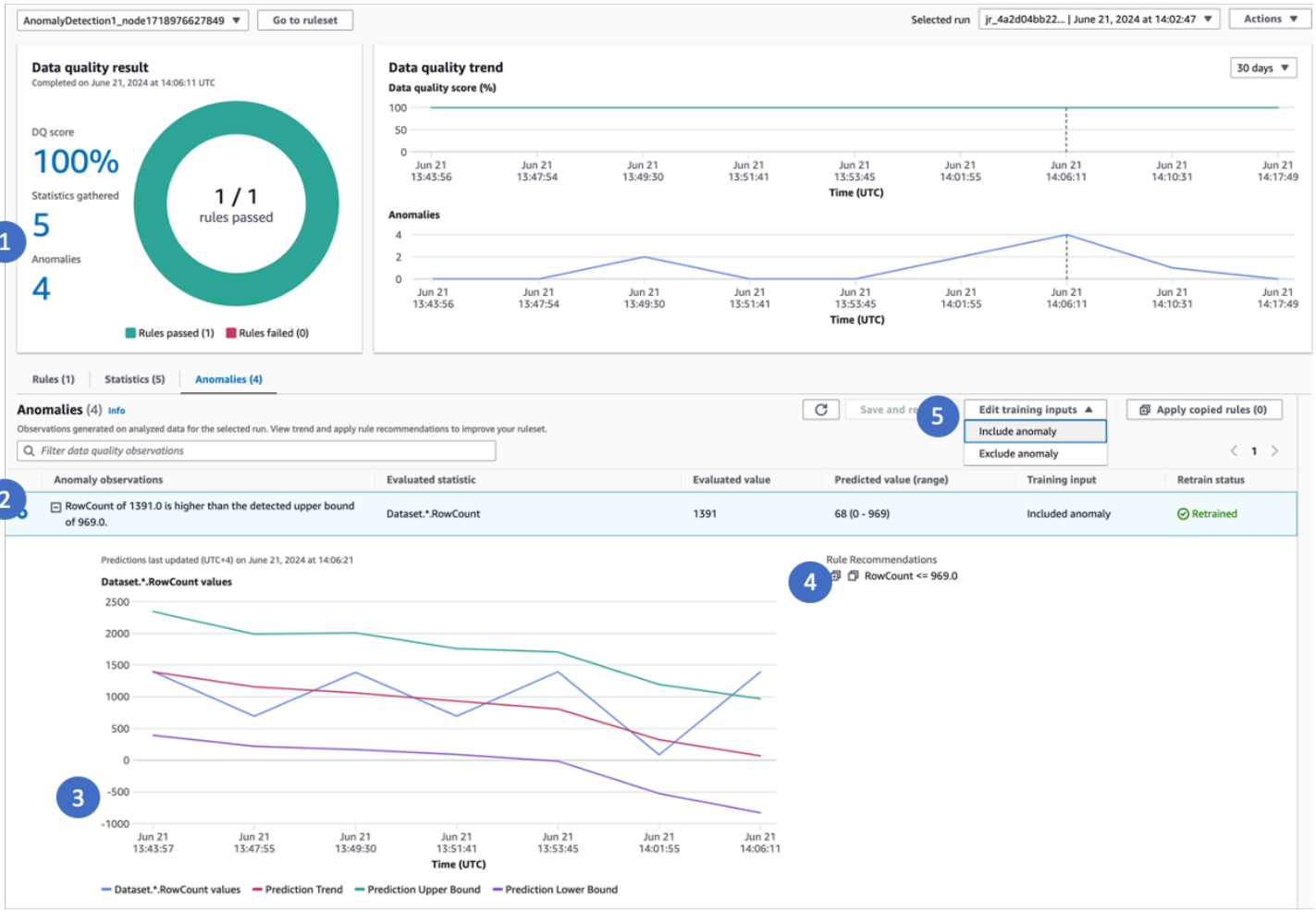


Die folgenden Komponenten auf der Registerkarte „Datenqualität“ bieten nützliche Informationen.

1. Wählen Sie die Registerkarte Datenqualität, um Messwerte zur Datenqualität anzuzeigen.

2. Wählen Sie eine bestimmte Jobausführungs-ID aus, um den Datenqualitätswert anzuzeigen.
3. In diesem Bereich werden drei wichtige Informationen angezeigt. Sie können jede Tabelle auswählen, um zu bestimmten Tabellen zu navigieren, um Anomalien, Datenstatistiken oder Regeln anzuzeigen.
   - Bewertung der Datenqualität bei der Konfiguration von Regeln.
   - Anzahl der von Regeln und Analyzern gesammelten Statistiken.
   - Gesamtzahl der festgestellten Anomalien.
4. Dieses Trenddiagramm zeigt, wie sich die Datenqualität im Laufe der Zeit entwickelt. Sie können den Mauszeiger über den Trend bewegen und zu einem bestimmten Zeitpunkt wechseln, zu dem sich die Datenqualitätswerte verschlechtert haben.
5. Anomalie-Trends im Zeitverlauf zeigen Ihnen die Anzahl der im Laufe der Zeit erkannten Anomalien.
6. Tabs:
   - Die Registerkarte „Regeln“ ist die Standardregisterkarte, auf der eine Liste aller Regeln und des Status angezeigt wird. Evaluierte Regeln sind bei dynamischen Regeln nützlich, um den tatsächlichen Wert anzuzeigen, mit dem die Regel ausgewertet wurde.
   - Auf der Registerkarte „Statistik“ werden alle Statistiken aufgeführt, sodass Sie sich die Metriken und Trends im Zeitverlauf ansehen können.
   - Auf der Registerkarte „Anomalien“ wird die Liste der erkannten Anomalien angezeigt.

Anzeige von Anomalien und Training des Algorithmus zur Erkennung von Anomalien



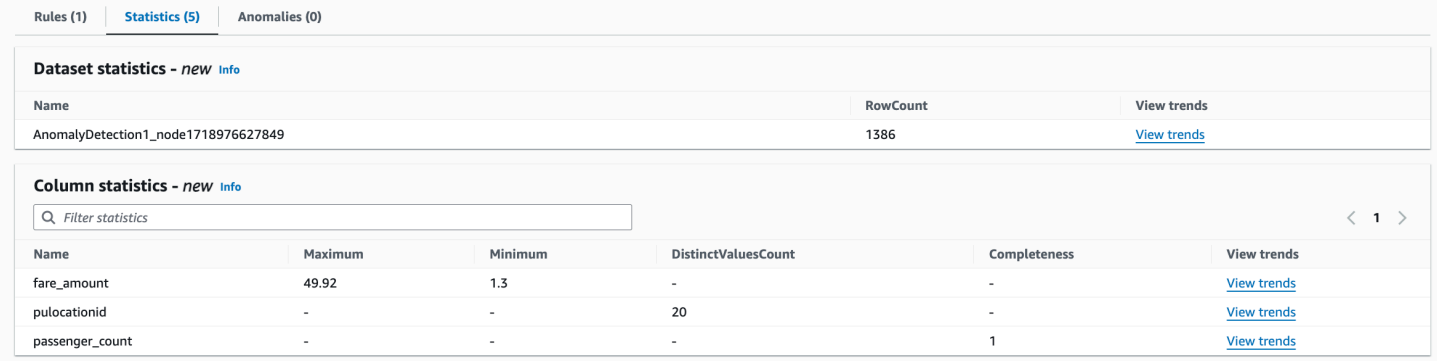
Bitte geben Sie das Bild oben an:

1. Wenn Anomalien erkannt werden, klicken Sie auf die Anomalie oder wählen Sie die Registerkarte Anomalien
2. AWS Glue Data Quality bietet eine detaillierte Erklärung der Anomalie, des tatsächlichen Werts und des prognostizierten Bereichs
3. AWS Glue Data Quality zeigt eine Trendlinie. Sie hat den tatsächlichen Wert, einen abgeleiteten Trend, der auf den tatsächlichen Werten (rote Linie), der Obergrenze und der Untergrenze basiert
4. AWS Glue Data Quality empfiehlt Datenqualitätsregeln, anhand derer die Muster für die future erfasst werden können. Sie können alle Regeln, die Ihnen empfohlen werden, kopieren und auf Ihren Datenqualitätsknoten anwenden, um diese Muster effektiv zu erfassen.
5. Sie können Eingaben für das Modell des maschinellen Lernens (ML) bereitstellen, um anomale Werte auszuschließen und so sicherzustellen, dass future Läufe Anomalien genau erkennen.

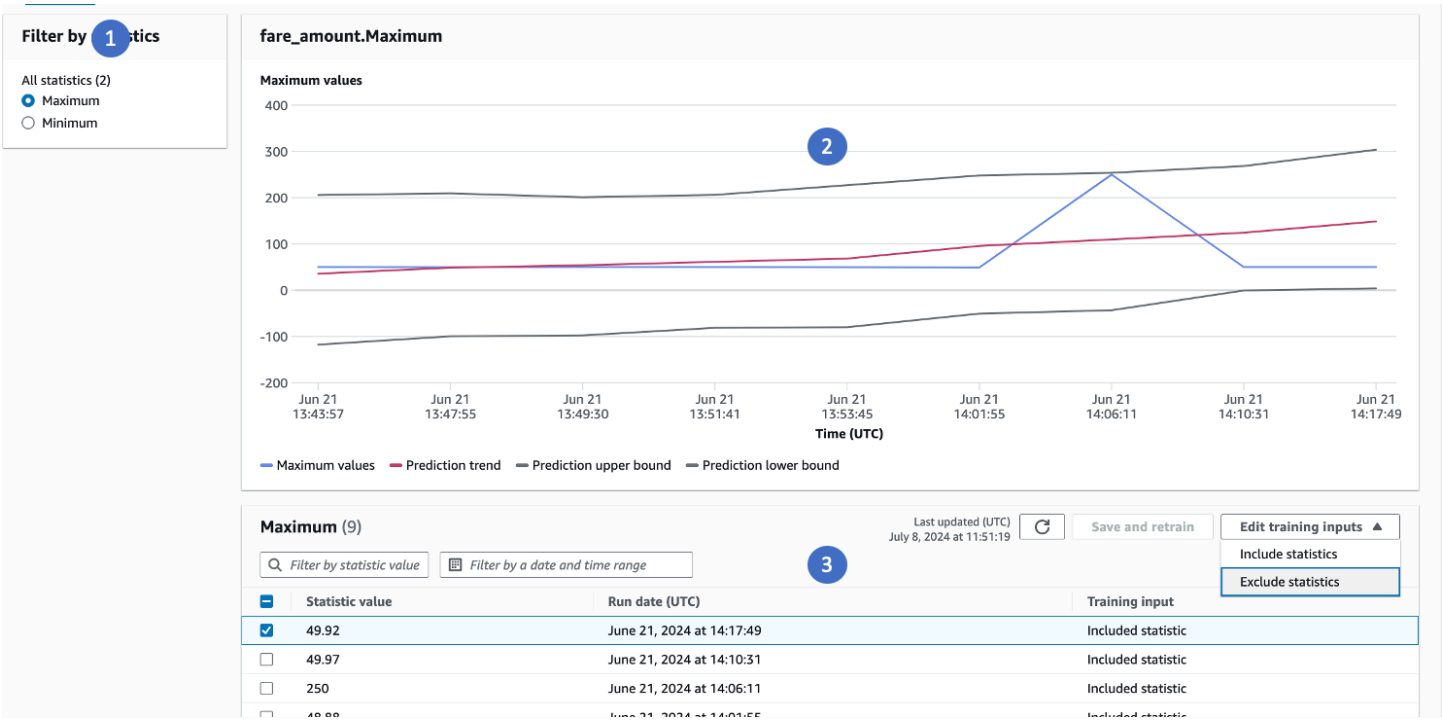
Wenn Sie Anomalien nicht explizit ausschließen, berücksichtigt AWS Glue Data Quality sie automatisch als Teil des Modells für future Prognosen. Es ist wichtig zu beachten, dass nur der letzte Lauf die von Ihnen eingegebenen Modelleingaben widerspiegelt. Wenn Sie beispielsweise zurückgegangen sind und anomale Punkte aus einigen vorherigen Durchläufen ausgeschlossen haben, spiegelt das Modell diese Änderungen nur wider, wenn Sie die Modelleingaben im letzten Lauf anzeigen und aktualisieren. Das Modell verwendet weiterhin die zuvor bereitgestellten Eingaben, bis Sie im letzten Durchlauf die erforderlichen Anpassungen vorgenommen haben. Indem Sie den Ausschluss anomaler Werte aktiv verwalten, können Sie das ML-Modell besser verstehen, was eine Anomalie für Ihre spezifischen Datenmuster und Anforderungen darstellt, was im Laufe der Zeit zu einer genaueren Erkennung von Anomalien führt.

Anzeige von Datenstatistiken im Zeitverlauf und Bereitstellung von Trainingsinformationen

Manchmal möchten Sie vielleicht Datenstatistiken oder Datenprofile einsehen und sehen, wie sie sich im Laufe der Zeit entwickeln. Wählen Sie dazu Statistik oder öffnen Sie den Tab Statistik. Sie können sich dann die neuesten Datenstatistiken ansehen, die von AWS Glue Data Quality gesammelt wurden.



Wenn Sie auf Trends anzeigen klicken, wird Ihnen angezeigt, wie sich die einzelnen Statistiken im Laufe der Zeit entwickeln.



1. Sie können die Statistik für eine bestimmte Spalte auswählen
2. Sie können sehen, wie sich die Trends entwickeln
3. Sie können ungewöhnliche Werte auswählen und diese ausschließen oder einbeziehen. Durch die Bereitstellung dieses Feedbacks schließt der Algorithmus die identifizierten anomalen Datenpunkte entweder aus oder schließt sie ein und trainiert das Modell erneut. Dieser Umschulungsprozess gewährleistet in Zukunft eine genaue Erkennung von Anomalien, da das Modell aus dem Feedback, das Sie gegeben haben, lernt, welche Werte als anomal angesehen werden sollten oder nicht.

   Durch diese Feedback-Schleife haben Sie die Möglichkeit, das Verständnis des Algorithmus dahingehend zu verfeinern, was eine Anomalie für Ihre spezifischen Datenmuster und Geschäftsanforderungen darstellt. Durch das Ausschließen von Werten, die nicht als Anomalien gekennzeichnet werden sollten, oder durch das Einbeziehen von Werten, die nicht als Anomalien gekennzeichnet werden sollten, kann das neu trainierte Modell besser zwischen erwarteten und wirklich anomalen Datenpunkten unterscheiden.

Datenqualität für ETL-Jobs in AWS Glue Studio-Notebooks

In diesem Tutorial erfahren Sie, wie Sie AWS Glue Data Quality für Extract, Transform, Load (ETL)-Aufträgen in AWS Glue Studio-Notebooks verwenden.

Sie können Notebooks in AWS Glue Studio verwenden, um Skripte für Aufträge zu bearbeiten und die Ausgabe anzuzeigen, ohne einen vollständigen Auftrag ausführen zu müssen. Sie können auch Markierungen hinzufügen und Notebooks als `.ipynb`-Dateien und Auftragsskripte speichern. Beachten Sie, dass Sie ein Notebook starten können, ohne Software lokal zu installieren oder Server zu verwalten. Wenn Sie mit Ihrem Code zufrieden sind, können Sie mithilfe von AWS Glue Studio Ihr Notebook ganz einfach in einen AWS Glue-Auftrag umwandeln.

Der Datensatz, den Sie in diesem Beispiel verwenden, besteht aus Zahlungsdaten von Medicare Providern, die aus zwei Data.cms.gov-Datensätzen heruntergeladen wurden: „Zusammenfassung der Anbieter von stationären potenziellen Zahlungsdienstleistern für die 100 wichtigsten diagnostizbezogenen Gruppen — 011" und „Stationäre Gebührendaten für das Geschäftsjahr 2011“. FY2

Nach dem Download änderten wir die Daten derart, dass wir einige fehlerhafte Datensätze am Ende der Datei hinzufügten. Diese geänderte Datei ist in einem öffentlichen Amazon S3 Bucket unter `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv` gespeichert.

Voraussetzungen

- AWS Glue-Rolle mit Amazon-S3-Berechtigung zum Schreiben in Ihren Ziel-Amazon S3-Bucket
- Ein neues Notebook (Informationen unter Erste Schritte mit Notebooks in AWS Glue Studio)

Erstellen eines ETL-Auftrags in AWS Glue Studio

So erstellen Sie einen ETL-Auftrag

1. Ändern Sie die Version der Sitzung auf AWS Glue 3.0.

   Entfernen Sie dazu alle Boilerplate-Codezellen mit dem folgenden Magic und führen Sie die Zelle aus. Beachten Sie, dass dieser Boilerplate-Code automatisch in der ersten Zelle bereitgestellt wird, wenn ein neues Notebook erstellt wird.

   ```
   %glue_version 3.0
   ```

2. Kopieren Sie den folgenden Code und führen Sie ihn in der Zelle aus.

   ```
   import sys
   from awsglue.transforms import *
   ```

```
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
```

3. Importieren Sie in der nächsten Zelle die `EvaluateDataQuality`-Klasse, die die AWS Glue Data Quality auswertet.

```
from awsgluedq.transforms import EvaluateDataQuality
```

4. Lesen Sie in der nächsten Zelle die Quelldaten mithilfe der CSV-Datei ein, die im öffentlichen Amazon-S3-Bucket gespeichert ist.

```
medicare = spark.read.format(
"csv").option(
"header", "true").option(
"inferSchema", "true").load(
's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
medicare.printSchema()
```

5. Konvertieren Sie AWS Glue DynamicFrame die Daten in eine.

```
from awsglue.dynamicframe import DynamicFrame
medicare_dyf = DynamicFrame.fromDF(medicare,glueContext,"medicare_dyf")
```

6. Erstellen Sie den Regelsatz mithilfe der Data Quality Definition Language (DQDL).

```
EvaluateDataQuality_ruleset = """
    Rules = [
        ColumnExists "Provider Id",
        IsComplete "Provider Id",
        ColumnValues  " Total Discharges " > 15
]
    ]
```

```
"""
```

7. Validieren Sie den Datensatz anhand des Regelsatzes.

```
EvaluateDataQualityMultiframe = EvaluateDataQuality().process_rows(
    frame=medicare_dyf,
    ruleset=EvaluateDataQuality_ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",
        "enableDataQualityCloudWatchMetrics": False,
        "enableDataQualityResultsPublishing": False,
    },
    additional_options={"performanceTuning.caching": "CACHE_NOTHING"},
)
```

8. Überprüfen Sie die Ergebnisse.

```
ruleOutcomes = SelectFromCollection.apply(
    dfc=EvaluateDataQualityMultiframe,
    key="ruleOutcomes",
    transformation_ctx="ruleOutcomes",
)

ruleOutcomes.toDF().show(truncate=False)
```

Ausgabe:

```
-------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|Rule                                 |Outcome|FailureReason
                  |EvaluatedMetrics                           |
+-------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|ColumnExists "Provider Id"           |Passed |null
                  |{}                                         |
|IsComplete "Provider Id"             |Passed |null
                  |{Column.Provider Id.Completeness -> 1.0}   |
```

```
|ColumnValues " Total Discharges " > 15|Failed |Value: 11.0 does not meet the
 constraint requirement!|{Column. Total Discharges .Minimum -> 11.0}|
+-------------------------------------+-------
+--------------------------------------------------
+-----------------------------------------+
```

9. Filtern Sie bestandene Zeilen und zeigen Sie die fehlgeschlagenen Zeilen in den Ergebnissen auf Zeilenebene der Data Quality an.

```
owLevelOutcomes = SelectFromCollection.apply(
dfc=EvaluateDataQualityMultiframe,
key="rowLevelOutcomes",
transformation_ctx="rowLevelOutcomes",
)

rowLevelOutcomes_df = rowLevelOutcomes.toDF() # Convert Glue DynamicFrame to
 SparkSQL DataFrame
rowLevelOutcomes_df_passed =
 rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Passed") # Filter only the Passed records.
rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Failed").show(5, truncate=False) # Review the Failed records
```

Ausgabe:

```
+----------------------------------------+----------
+-----------------------------------+-------------------------+------------
+--------------+-----------------+-----------------------------------
+------------------+-------------------------+----------------------
+-------------------------+-------------------------
+-----------------------------------------+---------------------------
+---------------------------+
|DRG Definition                            |Provider Id|Provider Name
           |Provider Street Address   |Provider City|Provider State|Provider Zip
 Code|Hospital Referral Region Description| Total Discharges | Average Covered
 Charges | Average Total Payments |Average Medicare Payments|DataQualityRulesPass
      |DataQualityRulesFail                    |DataQualityRulesSkip        |
DataQualityEvaluationResult|
+----------------------------------------+----------
+-----------------------------------+-------------------------+------------
+--------------+-----------------+-----------------------------------
```

```
+-----------------+-------------------------+----------------------
+-------------------------+-------------------------
+----------------------------------------+--------------------------
+--------------------------+
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10005      |MARSHALL MEDICAL CENTER SOUTH
        |2505 U S HIGHWAY 431 NORTH|BOAZ         |AL            |35957
 |AL - Birmingham                      |14               |$15131.85
  |$5787.57               |$4976.71                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10046      |RIVERVIEW REGIONAL MEDICAL
 CENTER    |600 SOUTH THIRD STREET    |GADSDEN      |AL            |35901
   |AL - Birmingham                      |14               |$67327.92
   |$5461.57               |$4493.57                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10083      |SOUTH BALDWIN REGIONAL
 MEDICAL CENTER|1613 NORTH MCKENZIE STREET|FOLEY        |AL            |36535
        |AL - Mobile                          |15               |$25411.33
         |$5282.93               |$4383.73                 |[IsComplete "Provider
 Id"]|[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
                   |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30002      |BANNER GOOD SAMARITAN MEDICAL
 CENTER |1111 EAST MCDOWELL ROAD   |PHOENIX      |AZ            |85006
 |AZ - Phoenix                         |11               |$34803.81
  |$7768.90               |$6951.45                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30010      |CARONDELET ST  MARYS HOSPITAL
        |1601 WEST ST MARY'S ROAD  |TUCSON       |AZ            |85745
 |AZ - Tucson                          |12               |$35968.50
  |$6506.50               |$5379.83                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
+----------------------------------------+----------
+-----------------------------------+-------------------------+------------
+-------------+-----------------+-----------------------------------
+-----------------+-------------------------+----------------------
+-------------------------+-------------------------
+----------------------------------------+--------------------------
+--------------------------+
only showing top 5 rows
```

Beachten Sie, dass AWS Glue Data Quality vier neue Spalten (DataQualityRulesPass, DataQualityRulesFail DataQualityRulesSkip, und DataQualityEvaluationResult) hinzugefügt hat. Dies zeigt die Datensätze an, die bestanden haben, die Datensätze, die fehlgeschlagen sind, die Regeln, die für die Auswertung auf Zeilenebene übersprungen wurden, und die Gesamtergebnisse auf Zeilenebene.

10. Schreiben Sie die Ausgabe in einen Amazon-S3-Bucket, um die Daten zu analysieren und die Ergebnisse zu visualisieren.

```
#Write the Passed records to the destination.

glueContext.write_dynamic_frame.from_options(
       frame = rowLevelOutcomes_df_passed,
       connection_type = "s3",
       connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
       format = "parquet")
```

Referenz zu Data Quality Definition Language (DQDL)

Data Quality Definition Language (DQDL) ist eine domänenspezifische Sprache, mit der Sie Regeln für AWS Glue Data Quality definieren.

In diesem Handbuch werden wichtige DQDL-Konzepte vorgestellt, die Ihnen beim Verständnis der Sprache helfen. Es bietet auch eine Referenz für DQDL-Regeltypen mit Syntax und Beispielen. Bevor Sie dieses Handbuch verwenden, empfehlen wir Ihnen, sich mit AWS Glue Data Quality vertraut zu machen. Weitere Informationen finden Sie unter AWS Glue Qualität der Daten.

> **Note**
>
> DynamicRules werden nur unterstützt in AWS Glue ETL.

Inhalt

- DQDL-Syntax
  - Regelstruktur
  - Zusammengesetzte Regeln

    - So funktionieren zusammengesetzte Regeln
  - Ausdrücke
    - Schlüsselwörter für NULL, EMPTY und WHITESPACES_ONLY
    - Filterung mit der Where-Klausel
  - Dynamische Regeln
  - Analysatoren
  - Kommentare
- DQDL-Regeltypreferenz
  - AggregateMatch
  - ColumnCorrelation
  - ColumnCount
  - ColumnDataType
  - ColumnExists
  - ColumnLength
  - ColumnNamesMatchPattern
  - ColumnValues
  - Vollständigkeit
  - CustomSQL
  - DataFreshness
  - DatasetMatch
  - DistinctValuesCount
  - Entropie
  - IsComplete
  - IsPrimaryKey
  - IsUnique
  - Mean
  - ReferentialIntegrity
  - RowCount
  - RowCountMatch
  - StandardDeviation

- Summe
- SchemaMatch
- Eindeutigkeit
- UniqueValueRatio
- DetectAnomalies
- FileFreshness
- FileMatch
- FileUniqueness
- FileSize

DQDL-Syntax

Ein DQDL-Dokument unterscheidet zwischen Groß- und Kleinschreibung und enthält einen Regelsatz, der einzelne Datenqualitätsregeln zusammenfasst. Um einen Regelsatz zu erstellen, müssen Sie eine Liste mit dem Namen `Rules` (groß geschrieben) erstellen, die durch ein Paar eckiger Klammern begrenzt ist. Die Liste sollte eine oder mehrere durch Kommas getrennte DQDL-Regeln wie im folgenden Beispiel enthalten.

```
Rules = [
   IsComplete "order-id",
   IsUnique "order-id"
]
```

Regelstruktur

Die Struktur einer DQDL-Regel hängt vom Regeltyp ab. Allerdings passen DQDL-Regeln im Allgemeinen in das folgende Format.

```
<RuleType> <Parameter> <Parameter> <Expression>
```

`RuleType` ist der zwischen Groß- und Kleinschreibung zu unterscheidende Name des Regeltyps, den Sie konfigurieren möchten. Beispiel: `IsComplete`, `IsUnique` oder `CustomSql`. Die Regelparameter unterscheiden sich für jeden Regeltyp. Eine vollständige Referenz der DQDL-Regeltypen und ihrer Parameter finden Sie unter DQDL-Regeltypreferenz.

Zusammengesetzte Regeln

DQDL unterstützt die folgenden logischen Operatoren, die Sie zum Kombinieren von Regeln verwenden können. Diese Regeln werden als zusammengesetzte Regeln bezeichnet.

and

Der logische `and`-Operator ergibt genau dann `true`, wenn die Regeln, die er verbindet, `true` sind. Andernfalls führt die kombinierte Regel zu `false`. Jede Regel, die Sie mit dem `and`-Operator verbinden, muss in runde Klammern eingeschlossen werden.

Im folgenden Beispiel wird der `and`-Operator zum Kombinieren zweier DQDL-Regeln verwendet.

```
(IsComplete "id") and (IsUnique "id")
```

or

Der logische `or`-Operator ergibt genau dann `true`, wenn eine oder mehrere der Regeln, die er verbindet, `true` sind. Jede Regel, die Sie mit dem `or`-Operator verbinden, muss in runde Klammern eingeschlossen werden.

Im folgenden Beispiel wird der `or`-Operator zum Kombinieren zweier DQDL-Regeln verwendet.

```
(RowCount "id" > 100) or (IsPrimaryKey "id")
```

Sie können denselben Operator verwenden, um mehrere Regeln zu verbinden, daher ist die folgende Regelkombination zulässig.

```
(Mean "Star_Rating" > 3) and (Mean "Order_Total" > 500) and (IsComplete "Order_Id")
```

Sie können die logischen Operatoren zu einem einzigen Ausdruck kombinieren. Zum Beispiel:

```
(Mean "Star_Rating" > 3) and ((Mean "Order_Total" > 500) or (IsComplete "Order_Id"))
```

Sie können auch komplexere, verschachtelte Regeln erstellen.

```
(RowCount > 0) or ((IsComplete "colA") and (IsUnique "colA"))
```

So funktionieren zusammengesetzte Regeln

Standardmäßig werden Verbundregeln als einzelne Regeln für den gesamten Datensatz oder die gesamte Tabelle ausgewertet, und anschließend werden die Ergebnisse kombiniert. Mit anderen Worten, zuerst wird die gesamte Spalte ausgewertet und dann der Operator angewendet. Dieses Standardverhalten wird im Folgenden anhand eines Beispiels erklärt:

```
# Dataset

+------+------+
|myCol1|myCol2|
+------+------+
|     2|     1|
|     0|     3|
+------+------+

# Overall outcome

+-------------------------------------------------------------+-------+
|Rule                                                         |Outcome|
+-------------------------------------------------------------+-------+
|(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)|Failed |
+-------------------------------------------------------------+-------+
```

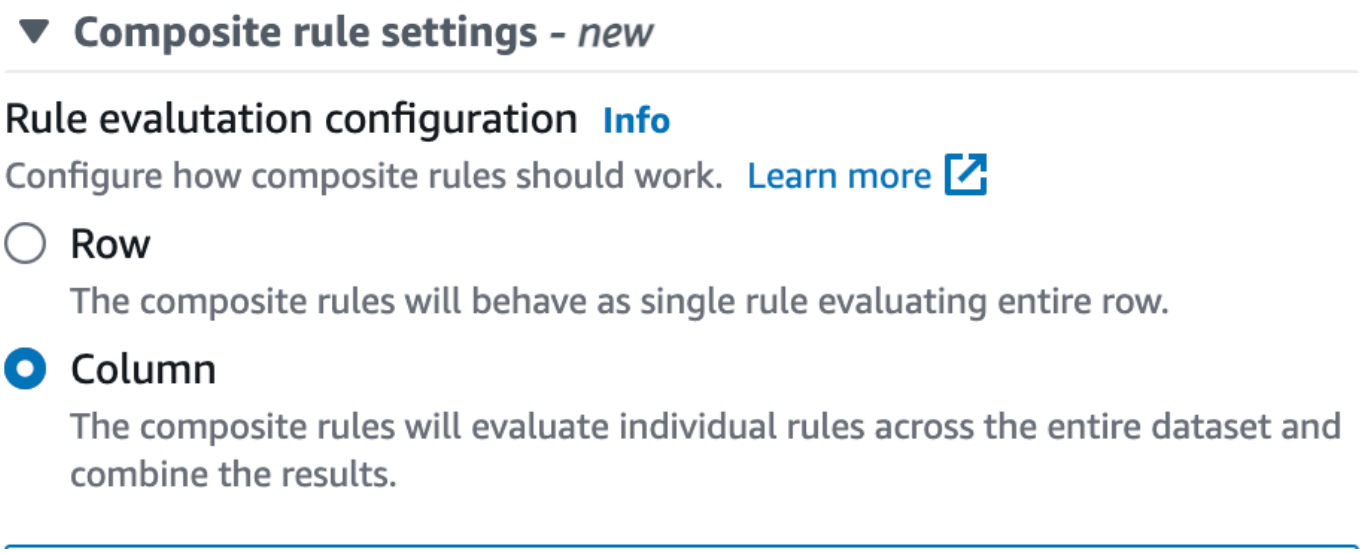
Im obigen Beispiel AWS Glue Data Quality bewertet zunächst(`ColumnValues "myCol1" > 1`), was zu einem Ausfall führen wird. Dann wird bewertet(`ColumnValues "myCol2" > 2`), was ebenfalls fehlschlagen wird. Die Kombination beider Ergebnisse wird als FEHLGESCHLAGEN vermerkt.

Wenn Sie jedoch ein SQL-ähnliches Verhalten bevorzugen, bei dem die gesamte Zeile ausgewertet werden muss, müssen Sie den `ruleEvaluation.scope` Parameter explizit festlegen, wie `additionalOptions` im folgenden Codeausschnitt gezeigt.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()

  val ruleset = """
```

```
    Rules = [
        (ColumnValues "age" >= 26) OR (ColumnLength "name" >= 4)
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "compositeRuleEvaluation.method":"ROW"
        }
      """
    )
  )
}
```

In AWS Glue Data Catalog können Sie diese Option einfach in der Benutzeroberfläche konfigurieren, wie unten gezeigt.



Sobald sie festgelegt sind, verhalten sich die zusammengesetzten Regeln wie eine einzige Regel, die die gesamte Zeile auswertet. Das folgende Beispiel veranschaulicht dieses Verhalten.

```
# Row Level outcome
```

```
+------+------+-----------------------------------------------------------------
+---------------------------+
|myCol1|myCol2|DataQualityRulesPass                                             |
DataQualityEvaluationResult|
+------+------+-----------------------------------------------------------------
+---------------------------+
|2     |1     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
                |
|0     |3     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
                |
+------+------+-----------------------------------------------------------------
+---------------------------+
```

Einige Regeln können in dieser Funktion nicht unterstützt werden, da ihr Gesamtergebnis von Schwellenwerten oder Verhältnissen abhängt. Sie sind unten aufgeführt.

Regeln, die sich auf Kennzahlen stützen:

- Vollständigkeit
- DatasetMatch
- ReferentialIntegrity
- Eindeutigkeit

Regeln, die von Schwellenwerten abhängen:

Wenn die folgenden Regeln einen Schwellenwert enthalten, werden sie nicht unterstützt. Regeln, die das nicht beinhalten, werden jedoch `with threshold` weiterhin unterstützt.

- ColumnDataType
- ColumnValues
- CustomSQL

Ausdrücke

Wenn ein Regeltyp keine boolesche Antwort erzeugt, müssen Sie einen Ausdruck als Parameter angeben, um eine boolesche Antwort zu erstellen. Die folgende Regel prüft beispielsweise den Mittelwert (Durchschnitt) aller Werte in einer Spalte anhand eines Ausdrucks, um ein wahres oder falsches Ergebnis zurückzugeben.

```
Mean "colA" between 80 and 100
```

Einige Regeltypen wie `IsUnique` und `IsComplete` geben bereits eine boolesche Antwort zurück.

In der folgenden Tabelle sind Ausdrücke aufgeführt, die Sie in DQDL-Regeln verwenden können.

Unterstützte DQDL-Ausdrücke

Expression	Beschreibung	Beispiel
=*x*	Löst den Wert auf, `true` wenn der Regeltyp der Antwort gleich ist. *x*	`Completeness "colA" = "1.0", ColumnValues "colA" = "2022-06-30"`
!=*x*	x Wird als wahr aufgelöst, wenn die Regeltypantwort ungleich ist. *x*	`ColumnValues "colA" != "a", ColumnValues "colA" != "2022-06-30"`
> *x*	Löst den Wert auf, `true` wenn die Regeltypantwort größer als ist. *x*	`ColumnValues "colA" > 10`
< *x*	Löst das Problem auf, `true` wenn die Regeltypantwort kleiner als ist. *x*	`ColumnValues "colA" < 1000, ColumnValues "colA" < "2022-06-30"`
>= *x*	Löst auf, `true` ob die Regeltypantwort größer oder gleich ist. *x*	`ColumnValues "colA" >= 10`
<= *x*	Löst auf, `true` ob die Regeltypantwort kleiner oder gleich ist. *x*	`ColumnValues "colA" <= 1000`
zwischen und *x* *y*	Löst sich zu `true` auf, wenn die Antwort des Regeltyps in einen bestimmten Bereich	`Mean "colA" between 8 and 100,`

Expression	Beschreibung	Beispiel
	fällt (exklusiv). Verwenden Sie diesen Ausdruckstyp nur für Zahlen- und Datumstypen.	`ColumnValues "colA" between "2022-05-31" and "2022-06-30"`
nicht zwischen *x* und *y*	Wird als wahr aufgelöst, wenn die Regeltyp-Antwort nicht in einen bestimmten Bereich (einschließlich) fällt. Sie sollten diesen Ausdrucks typ nur für Zahlen- und Datumstypen verwenden.	`ColumnValues "colA" not between "2022-05-31" and "2022-06-30"`
in [] *a, b, c, ...*	Löst sich zu `true` auf, wenn die Antwort des Regeltyps in der angegebenen Menge enthalten ist.	`ColumnValues "colA" in [ 1, 2, 3 ], ColumnValues "colA" in [ "a", "b", "c" ]`
nicht in [*a, b, c, ...*]	Löst das Problem auf, `true` wenn die Antwort vom Regeltyp nicht im angegeben en Satz enthalten ist.	`ColumnValues "colA" not in [ 1, 2, 3 ], ColumnValues "colA" not in [ "a", "b", "c" ]`
stimmt überein */ab+c/i*	Löst sich zu `true` auf, wenn die Antwort des Regeltyps mit einem regulären Ausdruck übereinstimmt.	`ColumnValues "colA" matches "[a-zA-Z]*"`
keine Spiele */ab+c/i*	Löst das Problem auf, `true` wenn die Regeltyp-Antwort keinem regulären Ausdruck entspricht.	`ColumnValues "colA" not matches "[a-zA-Z]*"`
`now()`	Funktioniert nur mit dem `ColumnValues` -Regeltyp zum Erstellen eines Datumsausdrucks.	`ColumnValues "load_date" > (now() - 3 days)`

Expression	Beschreibung	Beispiel

matches/in […]/not matches/notin [...] `with threshold`	Gibt den Prozentsatz der Werte an, die den Regelbedingungen entsprechen. Funktioniert nur mit den `CustomSQL` Regeltypen `ColumnValues` `ColumnDataType` , und.	`ColumnValues "colA" in ["A", "B"] with threshold > 0.8,`
		`ColumnValues "colA" matches "[a-zA-Z]*" with threshold between 0.2 and 0.9`
		`ColumnDataType "colA" = "Timestamp" with threshold > 0.9`

Schlüsselwörter für NULL, EMPTY und WHITESPACES_ONLY

Wenn Sie überprüfen möchten, ob eine Zeichenkettenspalte eine Null, eine leere Spalte oder eine Zeichenfolge mit nur Leerzeichen enthält, können Sie die folgenden Schlüsselwörter verwenden:

- NULL/null — Dieses Schlüsselwort wird für einen `null` Wert in einer Zeichenkettenspalte als wahr aufgelöst.

  `ColumnValues "colA" != NULL with threshold > 0.5`würde true zurückgeben, wenn mehr als 50% Ihrer Daten keine Nullwerte enthalten.

  `(ColumnValues "colA" = NULL) or (ColumnLength "colA" > 5)`würde für alle Zeilen, die entweder einen Nullwert oder eine Länge von > 5 haben, true zurückgeben. Beachten Sie, dass dafür die Option „compositeRuleEvaluation.method“ = „ROW“ verwendet werden muss.
- EMPTY/empty — Dieses Schlüsselwort wird für einen leeren Zeichenkettenwert („“) in einer Zeichenkettenspalte als wahr aufgelöst. Bei einigen Datenformaten werden Nullen in einer Zeichenkettenspalte in leere Zeichenketten umgewandelt. Dieses Schlüsselwort hilft dabei, leere Zeichenketten in Ihren Daten herauszufiltern.

  `(ColumnValues "colA" = EMPTY) or (ColumnValues "colA" in ["a", "b"])`würde true zurückgeben, wenn eine Zeile entweder leer ist, „a“ oder „b“. Beachten Sie, dass dies die Verwendung der Option „compositeRuleEvaluation.method“ = „ROW“ erfordert.
- WHITESPACES_ONLY//whitespaces_only — Dieses Schlüsselwort wird für eine Zeichenfolge mit nur Leerzeichen („“) in einer Zeichenkettenspalte als wahr aufgelöst.

`ColumnValues "colA" not in ["a", "b", WHITESPACES_ONLY]`würde true zurückgeben, wenn eine Zeile weder „a“ oder „b“ noch nur Leerzeichen enthält.

Unterstützte Regeln:

- ColumnValues

Wenn Sie für einen numerischen oder datumsbasierten Ausdruck überprüfen möchten, ob eine Spalte eine Null enthält, können Sie die folgenden Schlüsselwörter verwenden.

- NULL/null — Dieses Schlüsselwort wird für einen Nullwert in einer Zeichenfolgenspalte als wahr aufgelöst.

  `ColumnValues "colA" in [NULL, "2023-01-01"]`würde true zurückgeben, wenn ein Datum in Ihrer Spalte entweder `2023-01-01` oder Null ist.

  `(ColumnValues "colA" = NULL) or (ColumnValues "colA" between 1 and 9)`würde für alle Zeilen, die entweder einen Nullwert oder Werte zwischen 1 und 9 haben, true zurückgeben. Beachten Sie, dass hierfür die Option „compositeRuleEvaluation.method“ = „ROW“ verwendet werden muss.

  Unterstützte Regeln:

  - ColumnValues

Filterung mit der Where-Klausel

> **Note**
>
> Where Clause wird nur in AWS Glue 4.0 unterstützt.

Sie können Ihre Daten beim Erstellen von Regeln filtern. Dies ist hilfreich, wenn Sie bedingte Regeln anwenden möchten.

```
<DQDL Rule> where "<valid SparkSQL where clause> "
```

Der Filter muss mit dem `where` Schlüsselwort angegeben werden, gefolgt von einer gültigen SparkSQL, die in Anführungszeichen eingeschlossen ist. `("")`

Wenn die Regel, der Sie die WHERE-Klausel hinzufügen möchten, zu einer Regel mit einem Schwellenwert hinzufügen möchten, sollte die WHERE-Klausel vor der Schwellenwertbedingung angegeben werden.

```
<DQDL Rule> where "valid SparkSQL statement>" with threshold <threshold condition>
```

Mit dieser Syntax können Sie Regeln wie die folgenden schreiben.

```
Completeness "colA" > 0.5 where "colB = 10"
ColumnValues "colB" in ["A", "B"] where "colC is not null" with threshold > 0.9
ColumnLength "colC" > 10 where "colD != Concat(colE, colF)"
```

Wir werden überprüfen, ob die SparkSQL SparkSQL-Anweisung gültig ist. Falls ungültig, schlägt die Regelauswertung fehl und wir geben eine an im folgenden `IllegalArgumentException` Format aus:

```
Rule <DQDL Rule> where "<invalid SparkSQL>" has provided an invalid where clause :
<SparkSQL Error>
```

Verhalten der Where-Klausel, wenn die Identifizierung von Fehlerdatensätzen auf Zeilenebene aktiviert ist

Mit AWS Glue Data Quality können Sie bestimmte Datensätze identifizieren, bei denen Fehler aufgetreten sind. Wenn wir eine WHERE-Klausel auf Regeln anwenden, die Ergebnisse auf Zeilenebene unterstützen, kennzeichnen wir die Zeilen, die durch die WHERE-Klausel herausgefiltert werden, als`Passed`.

Wenn Sie es vorziehen, die herausgefilterten Zeilen separat als zu kennzeichnen`SKIPPED`, können Sie `additionalOptions` für den ETL-Job Folgendes festlegen.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()

  val ruleset = """
    Rules = [
```

```
        IsComplete "att2" where "att1 = 'a'"
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "rowLevelConfiguration.filteredRowLabel":"SKIPPED"
        }
      """
    )
  )
}
```

Sehen Sie sich als Beispiel die folgende Regel und den folgenden Datenrahmen an:

```
IsComplete att2 where "att1 = 'a'"
```

id	att1	att2	Ergebnisse auf Zeilenebene (Standard)	Ergebnisse auf Zeilenebene (Option übersprungen)	Kommentare
1	a	f	BESTANDEN	BESTANDEN	
2	b	d	BESTANDEN	SKIPPED	Zeile ist herausgefiltert, `att1` da nicht `"a"`
3	a	Null	FEHLGESCHLAGEN	FEHLGESCHLAGEN	
4	a	f	BESTANDEN	BESTANDEN	
5	b	Null	BESTANDEN	SKIPPED	Zeile ist herausgef

id	att1	att2	Ergebnisse auf Zeilenebene (Standard)	Ergebnisse auf Zeilenebene (Option übersprungen)	Kommentare
					iltert, `att1` da nicht `"a"`
6	a	f	BESTANDEN	BESTANDEN	

Dynamische Regeln

> **Note**
>
> Dynamische Regeln werden nur in AWS Glue ETL unterstützt und nicht im AWS Glue Data Catalog.

Sie können jetzt dynamische Regeln erstellen, um aktuelle Metriken, die anhand Ihrer Regeln erstellt wurden, mit ihren historischen Werten zu vergleichen. Diese historischen Vergleiche werden durch die Verwendung des Operators `last()` in Ausdrücken ermöglicht. Beispielsweise ist die Regel `RowCount > last()` erfolgreich, wenn die Anzahl der Zeilen im aktuellen Durchlauf größer ist als die letzte vorherige Anzahl der Zeilen im selben Datensatz. `last()` verwendet ein optionales Argument mit einer natürlichen Zahl, das beschreibt, wie viele vorherige Metriken berücksichtigt werden sollen. Bei `last(k)` verweist `k >= 1` auf die letzten k Metriken.

- Wenn keine Datenpunkte verfügbar sind, gibt `last(k)` den Standardwert 0,0 zurück.
- Wenn weniger als k Metriken verfügbar sind, gibt `last(k)` alle vorherigen Metriken zurück.

Verwenden Sie zur Bildung von Ausdrücken `last(k)`, wobei `k > 1` eine Aggregationsfunktion erfordert, um mehrere historische Ergebnisse auf eine einzelne Zahl zu reduzieren. Beispielsweise prüft `RowCount > avg(last(5))`, ob die aktuelle Zeilenanzahl des Datensatzes grundsätzlich höher ist als der Durchschnitt der letzten fünf Zeilenanzahlen desselben Datensatzes. `RowCount > last(5)` gibt einen Fehler aus, weil die aktuelle Zeilenanzahl des Datensatzes nicht aussagekräftig mit einer Liste verglichen werden kann.

Unterstützte Aggregationsfunktionen:

- `avg`
- `median`
- `max`
- `min`
- `sum`
- `std` (Standardabweichung)
- `abs` (absoluter Wert)
- `index(last(k), i)` ermöglicht die Auswahl des `i`.-letzten Werts der letzten `k`. `i` hat einen Nullindex, `index(last(3), 0)` gibt also den neuesten Datenpunkt zurück und `index(last(3), 3)` führt zu einem Fehler, da es nur drei Datenpunkte gibt und wir versuchen, den viertletzten zu indizieren.

Beispielausdrücke

ColumnCorrelation

- `ColumnCorrelation "colA" "colB" < avg(last(10))`

DistinctValuesCount

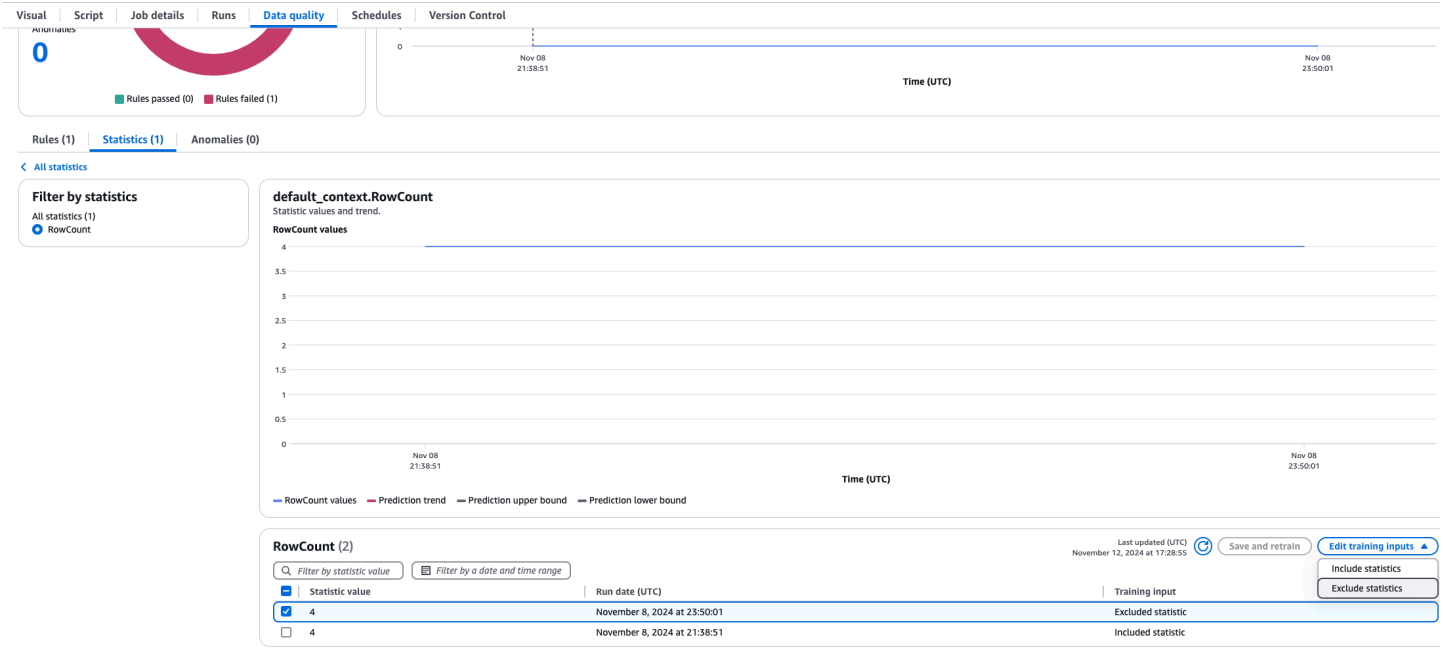
- `DistinctValuesCount "colA" between min(last(10))-1 and max(last(10))+1`

Die meisten Regeltypen mit numerischen Bedingungen oder Schwellenwerten unterstützen dynamische Regeln. In der bereitgestellten Tabelle Analysatoren und Regeln können Sie ermitteln, ob Ihr Regeltyp dynamische Regeln unterstützt.

Ausschließen von Statistiken aus dynamischen Regeln

Manchmal müssen Sie Datenstatistiken von Ihren dynamischen Regelberechnungen ausschließen. Nehmen wir an, Sie haben historische Daten geladen und möchten nicht, dass sich dies auf Ihre Durchschnittswerte auswirkt. Öffnen Sie dazu den Job in AWS Glue ETL und wählen Sie die Registerkarte Datenqualität. Wählen Sie dann Statistik und wählen Sie die Statistiken aus, die Sie ausschließen möchten. Sie werden ein Trenddiagramm zusammen mit einer Statistiktabelle sehen können. Wählen Sie die Werte aus, die Sie ausschließen möchten, und wählen Sie

Statistik ausschließen. Jetzt werden die ausgeschlossenen Statistiken nicht in die dynamischen Regelberechnungen einbezogen.



Analysatoren

> **Note**
>
> Analysatoren werden im AWS Glue Data Catalog nicht unterstützt.

DQDL-Regeln verwenden Funktionen, die als Analysatoren bezeichnet werden, um Informationen über Ihre Daten zu sammeln. Diese Informationen werden vom booleschen Ausdruck einer Regel verwendet, um zu bestimmen, ob die Regel erfolgreich sein oder fehlschlagen soll. Die RowCount Regel `RowCount > 5` verwendet beispielsweise einen Analysator für die Zeilenanzahl, um die Anzahl der Zeilen in Ihrem Datensatz zu ermitteln und diese Anzahl mit dem Ausdruck `> 5` zu vergleichen, um zu überprüfen, ob im aktuellen Datensatz mehr als fünf Zeilen vorhanden sind.

Manchmal empfehlen wir, anstelle von Regeln Analysatoren zu erstellen und diese dann Statistiken generieren zu lassen, anhand derer Anomalien erkannt werden können. In solchen Fällen können Sie Analysatoren erstellen. Analysatoren unterscheiden sich folgendermaßen von Regeln.

Merkmal	Analysatoren	Regeln
Teil des Regelsatzes	Ja	Ja
Generiert Statistiken	Ja	Ja
Generiert Beobachtungen	Ja	Ja
Kann eine Bedingung bewerten und durchsetzen	Nein	Ja
Aktionen wie das Stoppen von Aufträgen bei einem Fehler oder die Fortsetzung der Auftragsverarbeitung können konfiguriert werden	Nein	Ja

Analysatoren können unabhängig voneinander ohne Regeln existieren, sodass Sie sie schnell konfigurieren und schrittweise Regeln für die Datenqualität erstellen können.

Einige Regeltypen können in den `Analyzers`-Block Ihres Regelsatzes eingegeben werden, um die für Analysatoren erforderlichen Regeln auszuführen und Informationen zu sammeln, ohne auf Bedingungen zu prüfen. Einige Analysatoren sind nicht mit Regeln verknüpft und können nur in den `Analyzers`-Block eingegeben werden. Die folgende Tabelle gibt an, ob einzelne Elemente als Regel oder als eigenständige Analysatoren unterstützt werden und enthält zusätzliche Details zu jedem Regeltyp.

Beispiel für einen Regelsatz mit Analyzer

Der folgende Regelsatz verwendet

- eine dynamische Regel, mit der überprüft wird, ob ein Datensatz den Durchschnitt seiner letzten drei Auftragsausführungen überschreitet;
- einen `DistinctValuesCount`-Analysator, um die Anzahl unterschiedlicher Werte in der Spalte `Name` des Datensatzes zu erfassen;
- einen `ColumnLength`-Analysator, um die minimale und maximale Größe des `Name` im Laufe der Zeit nachzuverfolgen.

Die Ergebnisse der Analysatormetriken können auf der Registerkarte „Datenqualität“ der Auftragsausführung eingesehen werden.

```
Rules = [
   RowCount > avg(last(3))
]
Analyzers = [
   DistinctValuesCount "Name",
   ColumnLength "Name"
]
```

AWS Glue Data Quality unterstützt die folgenden Analysatoren.

Name des Analyzers	Funktionalität
`RowCount`	Berechnet die Zeilenanzahl für einen Datensatz
`Completeness`	Berechnet den Vollständigkeitsgrad einer Spalte
`Uniqueness`	Berechnet den Eindeutigkeitsprozentsatz einer Spalte
`Mean`	Berechnet den Mittelwert einer numerischen Spalte
`Sum`	Berechnet die Summe einer numerischen Spalte
`StandardDeviation`	Berechnet die Standardabweichung einer numerischen Spalte
`Entropy`	Berechnet die Entropie einer numerischen Spalte
`DistinctValuesCount`	Berechnet die Anzahl der unterschiedlichen Werte in einer Spalte

Name des Analyzers	Funktionalität
`UniqueValueRatio`	Berechnet das Verhältnis der Einzelwerte in einer Spalte
`ColumnCount`	Berechnet die Anzahl der Spalten in einem Datensatz
`ColumnLength`	Berechnet die Länge einer Spalte
`ColumnValues`	Berechnet das Minimum und Maximum für numerische Spalten. Berechnet das Minimum ColumnLength und das Maximum ColumnLen gth für nicht numerische Spalten
`ColumnCorrelation`	Berechnet Spaltenkorrelationen für gegebene Spalten
`CustomSql`	Berechnet Statistiken, die von CustomSQL zurückgegeben werden
`AllStatistics`	Berechnet die folgenden Statistiken: • RowCount, ColumnCount • Jede Spalte: Vollständigkeit, Einzigartigkeit • Numerisch: Min, Max, Entropie, Mittelwert, Standardwert, Summe • Zeichenfolge:, MinLength MaxLength

Kommentare

Sie können das Zeichen '#' verwenden, um Ihrem DQDL-Dokument einen Kommentar hinzuzufügen. Alles nach dem Zeichen '#' und bis zum Ende der Zeile wird von DQDL ignoriert.

```
Rules = [
    # More items should generally mean a higher price, so correlation should be
 positive
    ColumnCorrelation "price" "num_items" > 0
```

```
]
```

DQDL-Regeltypreferenz

Dieser Abschnitt enthält eine Referenz für jeden Regeltyp, den AWS Glue Data Quality unterstützt.

> **Note**
>
> - DQDL unterstützt derzeit keine verschachtelten oder listenförmigen Spaltendaten.
> - Die Werte in Klammern in der folgenden Tabelle werden durch die in den Regelargumenten angegebenen Informationen ersetzt.
> - Regeln erfordern in der Regel ein zusätzliches Argument für den Ausdruck.

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generiert Beobachtungen	Unterstützt die Syntax der Where-Klausel?
Aggregate Match	Überprüft, ob zwei Datensätze übereinstimmen, indem zusammenassende Metriken wie der Gesamtu	Eine oder mehrere Aggregationen	Wenn die Namen der ersten und zweiten Aggregationsspalte übereinstimmen: `Column.[C`	Ja	Nein	Nein	Nein	Nein	Nein

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generierung Beobachtungen	Unterstützt die Syntax der Where-Klausel?
	atz verglichen werden. Nützlich für Finanzurernehme um zu vergleichen, ob alle Daten aus Quellsysemen übernommen werden.		olumn].gregatetch Wenn die Namen der ersten und zweiten Aggregationsspalte nicht übereinstimmen: Column.[Column1,lumn2].gregatetch						

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?

AllStatis tics	Eigenstä diger Analysat r zum Sammeln mehrere Metriken für die angegeb e Spalte in einem Datensat .	Ein einziger Spaltenn me	Für Spalten jedes Typs:
			`Dataset .RowCou`
			`Column. [C olumn]. mpleter s`
			`Column. [C olumn]. iquenes`
			Zusätzlic he Metriken für Spalten mit Zeichenf lgewerte :
				Nein	Ja	Nein	Nein	Nein	Nein

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generiert Beobachtungen	Unterstützt die Syntax der Where-Klausel?

			`ColumnLength metrics`
			
			Zusätzliche Metriken für Spalten mit numerischen Werten:
			
			`ColumnValues metrics`						

ColumnCorrelation	Prüft, wie gut zwei Spalten korreliert sind.	Genau zwei Spaltennamen	`Multicolumn.[Column1,Column2].ColumnCorrelation`	Ja	Ja	Nein	Ja	Nein	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
ColumnC nt	Prüft, ob Spalten gelöscht wurden.	Keine	`Dataset .Columr unt`	Ja	Ja	Nein	Ja	Ja	Nein
ColumnD aType	Prüft, ob eine Spalte einem Datentyp entsprich t.	Genau ein Spaltenn me	`Column [C olumn]. lumnDat ype.Con iance`	Ja	Nein	Nein	Ja, im Schwelle wertausc uck auf Zeileneb ne	Nein	Ja

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generierte Beobachtungen	Unterstützt die Syntax der Where-Klausel?
ColumnExists	Prüft, ob Spalten in einem Datensatz vorhanden sind. Dadurch können Kunden Self-Service-Datenplattformen erstellen, um sicherzustellen, dass bestimmte Spalten verfügbar sind.	Genau ein Spaltenname	N/A	Ja	Nein	Nein	Nein	Nein	Nein

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generiert Beobachtungen	Unterstützt die Syntax der Where-Klausel?

ColumnLength	Prüft, ob die Datenlänge konsistent ist.	Genau ein Spaltenname	`Column.[Column].MaximumLength`
			
			`Column.[Column].MinimumLength`
			
			Zusätzliche Metrik, wenn ein Schwellenwert auf Zeilenebene angegeben ist:
			
			`Column.[Column].ColumnVal`
				Ja	Ja	Ja, wenn ein Schwellenwert auf Zeilenebene angegeben ist	Nein	Ja. Generiert Beobachtungen nur durch die Analyse der Mindest- und Maximallänge	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
			`s.Compl nce`						
ColumnN esMatchI ttern	Prüft, ob die Spaltenn men mit den definiert en Mustern übereins immen. Nützlich für Governa e-Teams, um die Konsiste z der Spaltenn men durchzus tzen.	Ein regulärer Ausdruck für Spaltenn men	`Dataset .ColumnesPatt nMatchR io`	Ja	Nein	Nein	Nein	Nein	Nein

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?

ColumnV ues	Prüft, ob die Daten gemäß den definiert en Werten konsister t sind. Diese Regel unterstüt zt reguläre Ausdrück .	Genau ein Spaltenn me	`Column [C olumn]. ximum`
			
			`Column [C olumn]. nimum`
			
			Zusätzlic he Metrik, wenn ein Schwelle wert auf Zeileneb ne angegeb ist:
			
			`Column [C olumn]. lumnVa s.Comp nce`
				Ja	Ja	Ja, wenn ein Schwelle wert auf Zeileneb ne angegeb ist	Nein	Ja. Generier Beobach ngen nur durch die Analyse der Mindest- und Maximal rte	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
Vollständ igkeit	Prüft, ob Daten leer sind oder NULLs sich darin befinden	Genau ein Spaltenn me	`Column [C olumn] mpleter s`	Ja	Ja	Ja	Ja	Ja	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?

CustomS	Kunden können nahezu jede Art von Datenqu itätsprüf ungen in SQL implemer ieren.	Eine SQL-Anwei sung
		
		(Optiona) Ein Schwelle wert auf Zeileneb ne
			`Dataset .Custon L`
			
			Zusätzlic he Metrik, wenn ein Schwelle wert auf Zeileneb ne angegeb ist:
			
			`Dataset .Custon L.Compl nce`
				Ja	Nein	Ja, wenn ein Schwelle wert auf Zeileneb ne angegeb ist	Ja	Nein	Nein

DataFres ness	Prüft, ob die Daten aktuell sind.	Genau ein Spaltenn me	`Column [C olumn]. taFresh ss.Comp ance`	Ja	Nein	Ja	Nein	Nein	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
DatasetM tch	Vergleich t zwei Datensät e und stellt fest, ob sie synchror sind.	Name eines Referenz atensatz s Eine Spaltenz weisung (Optiona) Nach Übereins immunge zu durchsuc ende Spalten	`Dataset [ Referer Dataset ias].Da setMatc`	Ja	Nein	Ja	Ja	Nein	Nein
DistinctV aluesCou t	Prüft auf doppelte Werte.	Genau ein Spaltenn me	`Column. [C olumn]. stinctV uesCour`	Ja	Ja	Ja	Ja	Ja	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
DetectAr malies	Überprüf die gemelde n Metriken eines anderen Regeltyp auf Anomalie .	Ein Regeltyp	Metrik(er), die vom Regeltyp rgument gemelde wurden	Ja	Nein	Nein	Nein	Nein	Nein
Entropie	Prüft die Daten auf Entropie.	Genau ein Spaltenn me	`Column [C olumn] tropy`	Ja	Ja	Nein	Ja	Nein	Ja
IsComple e	Prüft, ob 100 % der Daten vollständ ig sind.	Genau ein Spaltenn me	`Column [C olumn] mpleter s`	Ja	Nein	Ja	Nein	Nein	Ja

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generiert Beobachtungen	Unterstützt die Syntax der Where-Klausel?
IsPrimaryKey	Prüft, ob es sich bei einer Spalte um einen Primärschlüssel handelt (nicht NULL und eindeutig).	Genau ein Spaltenname	Für einzelne Spalte: `Column.[Column].Uniqueness` Für mehrere Spalten: `Multicolumn.[CommaDelimitedColumns].Uniqueness`	Ja	Nein	Ja	Nein	Nein	Ja
IsUnique	Prüft, ob 100 % der Daten eindeutig sind.	Genau ein Spaltenname	`Column.[Column].Uniqueness`	Ja	Nein	Ja	Nein	Nein	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstüt zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
Mean	Prüft, ob der Mittelwer t dem eingestel lten Schwelle wert entsprich t.	Genau ein Spaltenn me	`Column [C olumn] an`	Ja	Ja	Ja	Ja	Nein	Ja
Referent alIntegri ty	Prüft, ob zwei Datensät e referenzi elle Integritä t aufweise .	Ein oder mehrere Spaltenn men aus dem Datensat Ein oder mehrere Spaltenn men aus dem Referenz atensatz	`Column [R eferenc ataset/ as].Ref ential egrity`	Ja	Nein	Ja	Ja	Nein	Nein

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
RowCou	Prüft, ob die Anzahl der Datensät e einem Schwelle wert entsprich t.	Keine	`Dataset .RowCou`	Ja	Ja	Nein	Ja	Ja	Ja
RowCou atch	Prüft, ob die Datensat anzahl zwischer zwei Datensät en übereins immt.	Alias des Referenz atensatz s	`Dataset [ Referer Dataset ias].Rc ountMat`	Ja	Nein	Nein	Ja	Nein	Nein

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
Standarc eviation	Prüft, ob die Standarc bweichur dem Schwelle wert entsprich t.	Genau ein Spaltenn me	`Column [C olumn]. andardD iation`	Ja	Ja	Ja	Ja	Nein	Ja
Schemal ch	Prüft, ob das Schema zwischer zwei Datensät en übereins immt.	Alias des Referenz atensatz s	`Datase [ Refere Datase ias].Sc maMatc`	Ja	Nein	Nein	Ja	Nein	Nein

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
Summe	Prüft, ob die Summe einem festgeleg ten Schwelle wert entsprich t.	Genau ein Spaltenn me	`Column [C olumn] m`	Ja	Ja	Nein	Ja	Nein	Ja
Eindeutig keit	Prüft, ob die Eindeutig keit des Datensat es dem Schwelle wert entsprich t.	Genau ein Spaltenn me	`Column [C olumn] iquenes`	Ja	Ja	Ja	Ja	Nein	Ja

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?
UniqueV ueRatio	Prüft, ob das Verhältni s des Einzelwe ts dem Schwelle wert entsprich t.	Genau ein Spaltenn me	`Column [C olumn]. iqueVa Ratio`	Ja	Ja	Ja	Ja	Nein	Ja

FileFresh ness	Prüft, ob Dateien in Amazon S3 aktuell sind.	Datei- oder Ordnerpf d und ein Schwelle wert.	`Dataset .FileFr hness.C pliance`
			
			`Dataset .FileCo t`
				Ja	Nein	Nein	Nein	Nein	Nein

Regeltyp	Beschreibung	Argumente	Gemeldete Metriken	Als Regel unterstützt?	Als Analysator unterstützt?	Gibt Ergebnisse auf Zeilenebene zurück?	Unterstützung dynamischer Regeln?	Generieren Beobachtungen	Unterstützt die Syntax der Where-Klausel?
FileMatch	Prüft, ob der Inhalt der Datei mit einer Prüfsumme oder einer anderen Datei übereinstimmt. Diese Regel verwendet Prüfsummen, um zu überprüfen, ob zwei Dateien identisch sind.	Quelldatei- oder Ordnerpfad und Zieldatei- oder -ordnerpfad.	Es werden keine Statistiken generiert.	Ja	Nein	Nein	Nein	Nein	Nein

Regeltyp	Beschrei ung	Argumer	Gemelde Metriken	Als Regel unterstüt zt?	Als Analysat r unterstüt zt?	Gibt Ergebnis e auf Zeileneb ne zurück?	Unterstü zung dynamis er Regeln?	Generier Beobach ngen	Unterstüt zt die Syntax der Where-Kla usel?

FileSize	Prüft, ob die Größe einer Datei einer angegeb en Bedingur entsprich t.	Datei- oder Ordnerpf d und Schwelle wert.	`Dataset .FileSi .Compli ce`
			`Dataset .FileCo t`
			`Dataset .Maximu ileSize`
			`Dataset .Minimu ileSize`
				Ja	Nein	Nein	Nein	Nein	Nein

FileUniqu eness	Überprüf anhand von Prüfsumr n, ob Dateien eindeutig sind.	Datei- oder Ordnerpf d und Schwelle wert.	`Dataset .FileUn uenessR io` `Dataset .FileCo t`	Ja	Nein	Nein	Nein	Nein	Nein

Themen

- AggregateMatch
- ColumnCorrelation

- ColumnCount
- ColumnDataType
- ColumnExists
- ColumnLength
- ColumnNamesMatchPattern
- ColumnValues
- Vollständigkeit
- CustomSQL
- DataFreshness
- DatasetMatch
- DistinctValuesCount
- Entropie
- IsComplete
- IsPrimaryKey
- IsUnique
- Mean
- ReferentialIntegrity
- RowCount
- RowCountMatch
- StandardDeviation
- Summe
- SchemaMatch
- Eindeutigkeit
- UniqueValueRatio
- DetectAnomalies
- FileFreshness
- FileMatch
- FileUniqueness
- FileSize

AggregateMatch

Überprüft das Verhältnis zweier Spaltenaggregationen anhand eines bestimmten Ausdrucks. Dieser Regeltyp funktioniert für mehrere Datensätze. Die beiden Spaltenaggregationen werden ausgewertet und ein Quotient wird gebildet, indem das Ergebnis der ersten Spaltenaggregation durch das Ergebnis der zweiten Spaltenaggregation dividiert wird. Das Verhältnis wird mit dem bereitgestellten Ausdruck verglichen, um eine boolesche Antwort zu erzeugen.

Syntax

Spaltenaggregation

```
AggregateMatch <AGG_OPERATION> (<OPTIONAL_REFERENCE_ALIAS>.<COL_NAME>)
```

- AGG_OPERATION – Der für die Aggregation zu verwendende Vorgang. Derzeit werden `sum` und `avg` unterstützt.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- OPTIONAL_REFERENCE_ALIAS – Dieser Parameter muss angegeben werden, wenn die Spalte aus einem Referenzdatensatz und nicht aus dem primären Datensatz stammt. Wenn Sie diese Regel im AWS Glue-Datenkatalog verwenden, muss Ihr Referenz-Alias das Format "haben<database_name>. <table_name>. <column_name>

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- COL_NAME – Der Name der Spalte, die aggregiert werden soll.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

Beispiel: Durchschnitt

```
"avg(rating)"
```

Beispiel: Summe

```
"sum(amount)"
```

Beispiel: Durchschnitt der Spalte im Referenzdatensatz

```
"avg(reference.rating)"
```

Regel

```
AggregateMatch <AGG_EXP_1> <AGG_EXP_2> <EXPRESSION>
```

- AGG_EXP_1 – Die Aggregation der ersten Spalte.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AGG_EXP_2 – Die Aggregation der zweiten Spalte.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Aggregieren einer Übereinstimmung mithilfe von Summe

Die folgende Beispielregel prüft, ob die Summe der Werte in der `amount`-Spalte genau der Summe der Werte in der `total_amount`-Spalte entspricht.

```
AggregateMatch "sum(amount)" "sum(total_amount)" = 1.0
```

Beispiel: Aggregieren einer Übereinstimmung mithilfe des Durchschnitts

Die folgende Beispielregel prüft, ob der Durchschnitt der Werte in der `ratings`-Spalte mindestens 90 % des Durchschnitts der Werte in der `ratings`-Spalte im `reference`-Datensatz entspricht. Der Referenzdatensatz wird als zusätzliche Datenquelle im ETL- oder Data-Catalog-Erlebnis bereitgestellt.

In AWS Glue ETL können Sie Folgendes verwenden:

```
AggregateMatch "avg(ratings)" "avg(reference.ratings)" >= 0.9
```

Im AWS Glue-Datenkatalog können Sie Folgendes verwenden:

```
AggregateMatch "avg(ratings)" "avg(database_name.tablename.ratings)" >= 0.9
```

Null-Verhalten

Die `AggregateMatch` Regel ignoriert Zeilen mit NULL-Werten bei der Berechnung der Aggregationsmethoden (Summe/Mittelwert). Zum Beispiel:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Der Mittelwert der Spalte `units` ist (0 + 20 + 40)/3 = 20. Die Zeilen 101 und 103 werden bei dieser Berechnung nicht berücksichtigt.

ColumnCorrelation

Prüft die Korrelation zwischen zwei Spalten anhand eines bestimmten Ausdrucks. AWS Glue Data Quality verwendet den Korrelationskoeffizienten von Pearson, um die lineare Korrelation zwischen zwei Spalten zu messen. Das Ergebnis ist eine Zahl zwischen -1 und 1, die die Stärke und Richtung der Beziehung misst.

Syntax

```
ColumnCorrelation <COL_1_NAME> <COL_2_NAME> <EXPRESSION>
```

- COL_1_NAME – Der Name der ersten Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- COL_2_NAME – Der Name der zweiten Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Spaltenkorrelation

Die folgende Beispielregel prüft, ob der Korrelationskoeffizient zwischen den Spalten `height` und `weight` stark positiv korreliert ist (Koeffizientenwert größer als 0,8).

```
ColumnCorrelation "height" "weight" > 0.8
```

```
ColumnCorrelation "weightinkgs" "Salary" > 0.8 where "weightinkgs > 40"
```

Beispiel für dynamische Regeln

- `ColumnCorrelation "colA" "colB" between min(last(10)) and max(last(10))`
- `ColumnCorrelation "colA" "colB" < avg(last(5)) + std(last(5))`

Null-Verhalten

Die `ColumnCorrelation` Regel ignoriert Zeilen mit `NULL` Werten bei der Berechnung der Korrelation. Zum Beispiel:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Die Zeilen 101 und 103 werden ignoriert und haben den `ColumnCorrelation` Wert 1,0.

ColumnCount

Überprüft die Spaltenanzahl des primären Datensatzes anhand eines bestimmten Ausdrucks. Im Ausdruck können Sie die Anzahl der Spalten oder einen Spaltenbereich mithilfe von Operatoren wie > und < angeben.

Syntax

```
ColumnCount <EXPRESSION>
```

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Numerische Überprüfung der Spaltenanzahl

Die folgende Beispielregel prüft, ob die Spaltenanzahl innerhalb eines bestimmten Bereichs liegt.

```
ColumnCount between 10 and 20
```

Beispiel für dynamische Regeln

- `ColumnCount >= avg(last(10))`
- `ColumnCount between min(last(10))-1 and max(last(10))+1`

ColumnDataType

Prüft, ob die Werte in einer bestimmten Spalte in Apache Spark in den angegebenen Typ umgewandelt werden können. Akzeptiert einen `with threshold`-Ausdruck zur Prüfung auf eine Teilmenge der Werte in der Spalte.

Syntax

```
ColumnDataType <COL_NAME> = <EXPECTED_TYPE>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Zeichenfolgentyp

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- EXPECTED_TYPE – Der erwartete Typ der Werte in der Spalte.

  Unterstützte Werte: Boolean, Date, Timestamp, Integer, Double, Float, Long

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

- EXPRESSION – Ein optionaler Ausdruck zur Angabe des Prozentsatzes der Werte, die vom erwarteten Typ sein sollen.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

Beispiel: Spaltendatentyp-Ganzzahlen als Zeichenfolgen

Die folgende Beispielregel prüft, ob die Werte in der angegebenen Spalte, die vom Typ string ist, als Ganzzahlen umgewandelt werden können.

```
ColumnDataType "colA" = "INTEGER"
```

Beispiel: Spaltendatentyp-Ganzzahlen als Zeichenfolgen prüfen, ob eine Teilmenge der Werte vorhanden ist

Die folgende Beispielregel prüft, ob mehr als 90% der Werte in der angegebenen Spalte, die vom Typ Zeichenfolge ist, als Ganzzahlen umgewandelt werden können.

```
ColumnDataType "colA" = "INTEGER" with threshold > 0.9
```

ColumnExists

Prüft, ob eine Spalte vorhanden ist.

Syntax

```
ColumnExists <COL_NAME>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp

Beispiel: Spalte ist vorhanden

Die folgende Beispielregel prüft, ob die Spalte mit dem Namen `Middle_Name` vorhanden ist.

```
ColumnExists "Middle_Name"
```

ColumnLength

Prüft, ob die Länge jeder Zeile in einer Spalte einem bestimmten Ausdruck entspricht.

Syntax

```
ColumnLength <COL_NAME><EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Zeichenfolge
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Länge der Spaltenzeile

Die folgende Beispielregel prüft, ob der Wert in jeder Zeile in der Spalte mit dem Namen `Postal_Code` 5 Zeichen lang ist.

```
ColumnLength "Postal_Code" = 5
ColumnLength "weightinkgs" = 2 where "weightinkgs > 10"
```

Nullverhalten

Die `ColumnLength` Regel behandelt `NULL` s als Zeichenketten mit einer Länge von 0. Für eine `NULL` Zeile:

```
ColumnLength "Postal_Code" > 4 # this will fail
```

```
ColumnLength "Postal_Code" < 6 # this will succeed
```

Die folgende zusammengesetzte Beispielregel bietet eine Möglichkeit, `NULL` Werte explizit als falsch einzustufen:

```
(ColumnLength "Postal_Code" > 4) AND (ColumnValues "Postal_Code" != NULL)
```

ColumnNamesMatchPattern

Überprüft, ob die Namen aller Spalten im primären Datensatz mit dem angegebenen regulären Ausdruck übereinstimmen.

Syntax

```
ColumnNamesMatchPattern <PATTERN>
```

- PATTERN – Das Muster, anhand dessen Sie die Datenqualitätsregel bewerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

Beispiel: Spaltennamen stimmen mit dem Muster überein

Die folgende Beispielregel prüft, ob alle Spalten mit dem Präfix „aws_“ beginnen

```
ColumnNamesMatchPattern "aws_.*"
ColumnNamesMatchPattern "aws_.*" where "weightinkgs > 10"
```

ColumnValues

Führt einen Ausdruck für die Werte in einer Spalte aus.

Syntax

```
ColumnValues <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Zulässige Werte

Die folgende Beispielregel prüft, ob sich jeder Wert in der angegebenen Spalte in einer Gruppe zulässiger Werte befindet (einschließlich Null, leer und Zeichenfolgen mit nur Leerzeichen).

```
ColumnValues "Country" in [ "US", "CA", "UK", NULL, EMPTY, WHITESPACES_ONLY ]
ColumnValues "gender" in ["F", "M"] where "weightinkgs < 10"
```

Beispiel: Regulärer Ausdruck

Die folgende Beispielregel prüft die Werte in einer Spalte anhand eines regulären Ausdrucks.

```
ColumnValues "First_Name" matches "[a-zA-Z]*"
```

Beispiel: Datumswerte

Die folgende Beispielregel prüft die Werte in einer Datumsspalte anhand eines Datumsausdrucks.

```
ColumnValues "Load_Date" > (now() - 3 days)
```

Beispiel: Numerische Werte

Die folgende Beispielregel prüft, ob die Spaltenwerte mit einer bestimmten numerischen Einschränkung übereinstimmen.

```
ColumnValues "Customer_ID" between 1 and 2000
```

Null-Verhalten

Für alle `ColumnValues` Regeln (außer `!=` und`NOT IN`) gilt, dass `NULL` Zeilen die Regel nicht erfüllen. Wenn die Regel aufgrund eines Nullwerts fehlschlägt, wird die Fehlerursache wie folgt angezeigt:

```
Value: NULL does not meet the constraint requirement!
```

Die folgende zusammengesetzte Beispielregel bietet eine Möglichkeit, `NULL` Werte explizit zuzulassen:

```
(ColumnValues "Age" > 21) OR (ColumnValues "Age" = NULL)
```

Negierte ColumnValues Regeln, die die `not in` Syntax `!=` und verwenden, gelten für `NULL` Zeilen. Zum Beispiel:

```
ColumnValues "Age" != 21
```

```
ColumnValues "Age" not in [21, 22, 23]
```

Die folgenden Beispiele bieten eine Möglichkeit, Werte explizit als falsch einzuschätzen NULL

```
(ColumnValues "Age" != 21) AND (ColumnValues "Age" != NULL)
```

```
ColumnValues "Age" not in [21, 22, 23, NULL]
```

Vollständigkeit

Prüft den Prozentsatz vollständiger (nicht null) Werte in einer Spalte anhand eines bestimmten Ausdrucks.

Syntax

```
Completeness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Prozentualer Nullwert

Die folgenden Beispielregeln prüfen, ob mehr als 95 Prozent der Werte in einer Spalte vollständig sind.

```
Completeness "First_Name" > 0.95
Completeness "First_Name" > 0.95 where "weightinkgs > 10"
```

Beispiel für dynamische Regeln

- `Completeness "colA" between min(last(5)) - 1 and max(last(5)) + 1`
- `Completeness "colA" <= avg(last(10))`

Nullverhalten

Hinweis zu CSV-Datenformaten: Leere Zeilen in CSV-Spalten können mehrere Verhaltensweisen aufweisen.

- Wenn es sich bei einer Spalte um einen `String` Typ handelt, wird die leere Zeile als leere Zeichenfolge erkannt, sodass die `Completeness` Regel nicht erfüllt wird.
- Wenn eine Spalte einen anderen Datentyp hat wie`Int`, wird die leere Zeile als `Completeness` Regel erkannt `NULL` und erfüllt sie nicht.

CustomSQL

Dieser Regeltyp wurde erweitert, um zwei Anwendungsfälle zu unterstützen:

- Führen Sie eine benutzerdefinierte SQL-Anweisung für einen Datensatz aus und prüfen Sie den Rückgabewert anhand eines bestimmten Ausdrucks.
- Führen Sie eine benutzerdefinierte SQL-Anweisung aus, in der Sie in Ihrer SELECT-Anweisung einen Spaltennamen angeben, den Sie mit einer Bedingung vergleichen, um Ergebnisse auf Zeilenebene zu erhalten.

Syntax

```
CustomSql <SQL_STATEMENT> <EXPRESSION>
```

- SQL_STATEMENT – Eine SQL-Anweisung, die einen einzelnen numerischen Wert zurückgibt, der von doppelten Anführungszeichen umgeben ist.
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Benutzerdefiniertes SQL zum Abrufen eines Gesamtregelergebnisses

Diese Beispielregel verwendet eine SQL-Anweisung, um die Datensatzanzahl für einen Datensatz abzurufen. Die Regel prüft dann, ob die Datensatzanzahl zwischen 10 und 20 liegt.

```
CustomSql "select count(*) from primary" between 10 and 20
```

Beispiel: Benutzerdefiniertes SQL zum Abrufen von Ergebnissen auf Zeilenebene

Diese Beispielregel verwendet eine SQL-Anweisung, bei der Sie in Ihrer SELECT-Anweisung einen Spaltennamen angeben, den Sie mit einer Bedingung vergleichen, um Ergebnisse auf Zeilenebene zu erhalten. Ein Ausdruck für eine Schwellenwertbedingung definiert einen Schwellenwert dafür, wie viele Datensätze fehlschlagen sollten, damit die gesamte Regel fehlschlägt. Beachten Sie, dass eine Regel nicht gleichzeitig eine Bedingung und ein Schlüsselwort enthalten darf.

```
CustomSql "select Name from primary where Age  > 18"
```

or

```
CustomSql "select Name from primary where Age > 18" with threshold  > 3
```

> **Important**
>
> Der `primary`-Alias steht stellvertretend für den Namen des Datensatzes, den Sie auswerten möchten. Wenn Sie mit visuellen ETL-Aufträgen in der Konsole arbeiten, steht `primary` immer für das `DynamicFrame`, das an die `EvaluateDataQuality.apply()`-Transformation übergeben wird. Wenn Sie den AWS Glue-Datenkatalog verwenden, um Datenqualitätsaufgaben für eine Tabelle auszuführen, `primary` stellt dies die Tabelle dar.

Wenn Sie dabei sind AWS Glue Im Datenkatalog können Sie auch die tatsächlichen Tabellennamen verwenden:

```
CustomSql "select count(*) from database.table" between 10 and 20
```

Sie können auch mehrere Tabellen miteinander verbinden, um verschiedene Datenelemente zu vergleichen:

```
CustomSql "select count(*) from database.table inner join database.table2 on id1 = id2"
 between 10 and 20
```

In AWS Glue ETL und CustomSQL können Datensätze identifizieren, die die Datenqualitätsprüfungen nicht bestanden haben. Damit dies funktioniert, müssen Sie Datensätze zurückgeben, die Teil der Primärtabelle sind, für die Sie die Datenqualität bewerten. Datensätze, die im Rahmen der Abfrage zurückgegeben werden, gelten als erfolgreich und Datensätze, die nicht zurückgegeben werden, gelten als fehlgeschlagen. Dies funktioniert, indem das Ergebnis Ihrer CustomSQL-Abfrage mit dem

ursprünglichen Datensatz verknüpft wird. Aufgrund der Komplexität Ihrer SQL-Abfrage kann es zu Leistungseinbußen kommen.

So gehen Sie vor:

- Sie müssen mindestens eine Spalte aus Ihrer Primärtabelle auswählen.
  - `select count(*) from primary`ist eine gültige Abfrage für die gesamte CustomSQL-DQ-Regel, aber nicht für Custom SQL auf Zeilenebene.
  - Diese Regel löst bei der Auswertung einen Fehler aus: `The output from CustomSQL must contain at least one column that matches the input dataset for AWS Glue Data Quality to provide row level results. The SQL query is a valid query but the columns from the SQL result are not present in the Input Dataset. Ensure that matching columns are returned from the SQL.`
- Wählen Sie in Ihrer SQL-Abfrage einen `Primärschlüssel` aus Ihrer Tabelle oder wählen Sie eine Gruppe von Spalten aus, die einen zusammengesetzten Schlüssel bilden. Andernfalls kann es zu inkonsistenten Ergebnissen kommen, da doppelte Zeilen übereinstimmen und die Leistung beeinträchtigt wird.
- Wählen Sie NUR Schlüssel aus Ihrer Primärtabelle und nicht aus Ihren Referenztabellen aus.

Die folgende Regel stellt sicher, dass Datensätze mit einem Alter von weniger als 100 Jahre als erfolgreich identifiziert werden und Datensätze darüber als fehlgeschlagen markiert werden.

```
CustomSql "select id from primary where age < 100"
```

Diese CustomSQL-Regel wird bestanden, wenn 50 % der Datensätze älter als 10 Jahre sind, und identifiziert auch Datensätze, die fehlgeschlagen sind. Die von diesem CustomSQL zurückgegebenen Datensätze gelten als bestanden, während die nicht zurückgegebenen Datensätze als fehlgeschlagen gelten.

```
CustomSQL "select ID, CustomerID from primary where age > 10" with threshold > 0.5
```

Hinweis: Die CustomSQL-Regel schlägt fehl, wenn Sie Datensätze zurückgeben, die im Datensatz nicht verfügbar sind.

DataFreshness

Prüft die Aktualität der Daten in einer Spalte, indem die Differenz zwischen der aktuellen Uhrzeit und den Werten einer Datumsspalte ausgewertet wird. Sie können einen zeitbasierten Ausdruck für diesen Regeltyp angeben, um sicherzustellen, dass die Spaltenwerte aktuell sind.

Syntax

```
DataFreshness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Datum
- AUSDRUCK – Ein numerischer Ausdruck in Stunden oder Tagen. Sie müssen die Zeiteinheit in Ihrem Ausdruck angeben.

Beispiel: Datenaktualität

Die folgenden Beispielregeln prüfen die Datenaktualität.

```
DataFreshness "Order_Date" <= 24 hours
DataFreshness "Order_Date" between 2 days and 5 days
```

Null-Verhalten

Die `DataFreshness` Regeln schlagen bei Zeilen mit `NULL` Werten fehl. Wenn die Regel aufgrund eines Nullwerts fehlschlägt, wird die Fehlerursache wie folgt angezeigt:

```
80.00 % of rows passed the threshold
```

wobei 20% der fehlgeschlagenen Zeilen die Zeilen mit enthaltenNULL.

Die folgende zusammengesetzte Beispielregel bietet eine Möglichkeit, `NULL` Werte explizit zuzulassen:

```
(DataFreshness "Order_Date" <= 24 hours) OR (ColumnValues "Order_Date" = NULL)
```

Datenaktualität für Amazon S3 S3-Objekte

Manchmal müssen Sie die Aktualität der Daten anhand der Erstellungszeit der Amazon S3 S3-Datei überprüfen. Zu diesem Zweck können Sie den folgenden Code verwenden, um den Zeitstempel abzurufen und ihn Ihrem Datenrahmen hinzuzufügen. Anschließend können Sie Datenaktualisierungsprüfungen durchführen.

```
df = glueContext.create_data_frame.from_catalog(database = "default", table_name =
 "mytable")
df = df.withColumn("file_ts", df["_metadata.file_modification_time"])

Rules = [
 DataFreshness "file_ts" < 24 hours
]
```

DatasetMatch

Prüft, ob die Daten im Primärdatensatz mit den Daten in einem Referenzdatensatz übereinstimmen. Die beiden Datensätze werden mithilfe der bereitgestellten Schlüsselspaltenzuordnungen verbunden. Wenn Sie die Gleichheit der Daten nur in diesen Spalten prüfen möchten, können zusätzliche Spaltenzuordnungen bereitgestellt werden. Beachten Sie, dass Ihre Join-Schlüssel eindeutig sein müssen und nicht NULL sein sollten (müssen ein Primärschlüssel sein), DataSetMatchdamit sie funktionieren. Wenn Sie diese Bedingungen nicht erfüllen, erhalten Sie die Fehlermeldung „Die bereitgestellte Schlüsselkarte ist für bestimmte Datenrahmen nicht geeignet“. In Fällen, in denen Sie keine eindeutigen Schlüssel verknüpft haben können, sollten Sie die Verwendung anderer Regeltypen in Betracht ziehen, z. B. AggregateMatchden Abgleich bei Übersichtsdaten.

Syntax

```
DatasetMatch <REFERENCE_DATASET_ALIAS> <JOIN CONDITION WITH
 MAPPING> <OPTIONAL_MATCH_COLUMN_MAPPINGS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS – Der Alias des Referenzdatensatzes, mit dem Sie Daten aus dem Primärdatensatz vergleichen.
- KEY_COLUMN_MAPPINGS – Eine durch Kommas getrennte Liste von Spaltennamen, die einen Schlüssel in den Datensätzen bilden. Wenn die Spaltennamen in beiden Datensätzen nicht identisch sind, müssen Sie sie durch einen `->` trennen
- OPTIONAL_MATCH_COLUMN_MAPPINGS – Sie können diesen Parameter angeben, wenn Sie nur in bestimmten Spalten nach übereinstimmenden Daten suchen möchten. Es verwendet dieselbe Syntax wie die Schlüsselspaltenzuordnungen. Wenn dieser Parameter nicht angegeben

wird, werden die Daten in allen verbleibenden Spalten abgeglichen. Die verbleibenden Nicht-Schlüsselspalten müssen in beiden Datensätzen dieselben Namen aufweisen.

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Zuordnen von Set-Datensätzen mithilfe der ID-Spalte

Die folgende Beispielregel prüft, ob mehr als 90 % des primären Datensatzes mit dem Referenzdatensatz übereinstimmt, und verwendet dabei die Spalte „ID“, um die beiden Datensätze zu verbinden. In diesem Fall werden alle Spalten verglichen.

```
DatasetMatch "reference" "ID" >= 0.9
```

Beispiel: Festlegen von Datensätzen mithilfe mehrerer Schlüsselspalten

Im folgenden Beispiel haben der Primärdatensatz und der Referenzdatensatz unterschiedliche Namen für die Schlüsselspalten. `ID_1` und `ID_2` bilden zusammen einen zusammengesetzten Schlüssel im Primärdatensatz. `ID_ref1` und `ID_ref2` bilden zusammen einen zusammengesetzten Schlüssel im Referenzdatensatz. In diesem Szenario können Sie die spezielle Syntax zur Bereitstellung der Spaltennamen verwenden.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_ref2->ID_ref2" >= 0.9
```

Beispiel: Festlegung von Datensätzen mithilfe mehrerer Schlüsselspalten und Überprüfung der Übereinstimmung bestimmter Spalten

Dieses Beispiel basiert auf dem vorherigen Beispiel. Wir möchten überprüfen, ob nur die Spalte mit den Beträgen übereinstimmt. Diese Spalte ist im Primärdatensatz `Amount1` benannt und im Referenzdatensatz `Amount2` benannt. Sie möchten eine exakte Übereinstimmung.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_2->ID_ref2" "Amount1->Amount2" >= 0.9
```

DistinctValuesCount

Prüft die Anzahl unterschiedlicher Werte in einer Spalte anhand eines bestimmten Ausdrucks.

Syntax

```
DistinctValuesCount <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Anzahl eindeutiger Spaltenwerte

Die folgende Beispielregel prüft, ob die Spalte mit dem Namen `State` mehr als 3 unterschiedliche Werte enthält.

```
DistinctValuesCount "State" > 3
DistinctValuesCount "Customer_ID" < 6  where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `DistinctValuesCount "colA" between avg(last(10))-1 and avg(last(10))+1`
- `DistinctValuesCount "colA" <= index(last(10),2) + std(last(5))`

Entropie

Prüft, ob der Entropie-Wert einer Spalte mit einem bestimmten Ausdruck übereinstimmt. Die Entropie misst den Informationsgehalt, der in einer Nachricht enthalten ist. Bei gegebener Wahrscheinlichkeitsverteilung über Werte in einer Spalte beschreibt die Entropie, wie viele Bits benötigt werden, um einen Wert zu identifizieren.

Syntax

```
Entropy <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Spaltenentropie

Die folgende Beispielregel prüft, ob die Spalte mit dem Namen `Feedback` einen Entropiewert größer als eins hat.

```
Entropy "Star_Rating" > 1
Entropy "First_Name" > 1  where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `Entropy "colA" < max(last(10))`
- `Entropy "colA" between min(last(10)) and max(last(10))`

IsComplete

Prüft, ob alle Werte in einer Spalte vollständig (nicht null) sind.

Syntax

```
IsComplete <COL_NAME>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp

Beispiel: Nullwerte

Das folgende Beispiel prüft, ob alle Werte in einer Spalte mit dem Namen `email` nicht null sind.

```
IsComplete "email"
IsComplete "Email" where "Customer_ID between 1 and 50"
IsComplete "Customer_ID"  where "Customer_ID < 16 and Customer_ID != 12"
IsComplete "passenger_count" where "payment_type<>0"
```

Nullverhalten

Hinweis zu CSV-Datenformaten: Leere Zeilen in CSV-Spalten können mehrere Verhaltensweisen aufweisen.

- Wenn es sich bei einer Spalte um einen `String` Typ handelt, wird die leere Zeile als leere Zeichenfolge erkannt, sodass die `Completeness` Regel nicht erfüllt wird.

- Wenn eine Spalte einen anderen Datentyp hat wie`Int`, wird die leere Zeile als `Completeness` Regel erkannt `NULL` und erfüllt sie nicht.

IsPrimaryKey

Prüft, ob eine Spalte einen Primärschlüssel enthält. Eine Spalte enthält einen Primärschlüssel, wenn alle Werte in der Spalte eindeutig und vollständig (nicht null) sind. Sie können auch nach Primärschlüsseln mit mehreren Spalten suchen.

Syntax

```
IsPrimaryKey <COL_NAME>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp

Beispiel: Primärschlüssel

Die folgende Beispielregel prüft, ob die Spalte mit dem Namen `Customer_ID` einen Primärschlüssel enthält.

```
IsPrimaryKey "Customer_ID"
IsPrimaryKey "Customer_ID" where "Customer_ID < 10"
```

Beispiel: Primärschlüssel mit mehreren Spalten. Alle folgenden Beispiele sind gültig.

```
IsPrimaryKey "colA" "colB"
IsPrimaryKey "colA" "colB" "colC"
IsPrimaryKey colA "colB" "colC"
```

IsUnique

Prüft, ob alle Werte in einer Spalte eindeutig sind, und gibt einen Booleschen Wert zurück.

Syntax

```
IsUnique <COL_NAME>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp

Beispiele

Die folgende Beispielregel prüft, ob alle Werte in einer Spalte mit dem Namen `email` eindeutig sind.

```
IsUnique "email"
IsUnique "Customer_ID" where "Customer_ID < 10"]
```

Die folgende Beispielregel überprüft mehrere Spalten.

```
IsUnique "vendorid" "tpep_pickup_datetime"
```

Mean

Prüft, ob der Mittelwert (Durchschnitt) aller Werte in einer Spalte mit einem bestimmten Ausdruck übereinstimmt.

Syntax

```
Mean <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Durchschnittswert

Die folgende Beispielregel prüft, ob der Durchschnitt aller Werte in einer Spalte einen Schwellenwert überschreitet.

```
Mean "Star_Rating" > 3
Mean "Salary" < 6200 where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `Mean "colA" > avg(last(10)) + std(last(2))`
- `Mean "colA" between min(last(5)) - 1 and max(last(5)) + 1`

Null-Verhalten

Die `Mean` Regel ignoriert Zeilen mit `NULL` Werten bei der Berechnung des Mittelwerts. Zum Beispiel:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Der Mittelwert der Spalte `units` ist (0 + 20 + 40)/3 = 20. Die Zeilen 101 und 103 werden bei dieser Berechnung nicht berücksichtigt.

ReferentialIntegrity

Überprüft, inwieweit die Werte einer Spaltengruppe im Primärdatensatz eine Teilmenge der Werte einer Spaltengruppe in einem Referenzdatensatz sind.

Syntax

```
ReferentialIntegrity <PRIMARY_COLS> <REFERENCE_DATASET_COLS> <EXPRESSION>
```

- PRIMARY_COLS – Eine durch Kommas getrennte Liste von Spaltennamen im primären Datensatz.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- REFERENCE_DATASET_COLS – Dieser Parameter besteht aus zwei durch einen Punkt getrennten Teilen. Der erste Teil ist der Alias des Referenzdatensatzes. Der zweite Teil ist die durch Kommas getrennte Liste der Spaltennamen im Referenzdatensatz in geschweiften Klammern.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Überprüfung der referenziellen Integrität einer PLZ-Spalte

Die folgende Beispielregel prüft, ob mehr als 90 % der Werte in der `zipcode`-Spalte im Primärdatensatz in der `zipcode`-Spalte im `reference`-Datensatz vorhanden sind.

```
ReferentialIntegrity "zipcode" "reference.zipcode" >= 0.9
```

Beispiel: Überprüfung der referenziellen Integrität der Spalten „Stadt“ und „Bundesland“

Im folgenden Beispiel sind im Primärdatensatz und im Referenzdatensatz Spalten mit Stadt- und Bundeslandinformationen vorhanden. Die Namen der Spalten sind in beiden Datensätzen unterschiedlich. Die Regel prüft, ob die Wertemenge der Spalten im Primärdatensatz genau mit der Wertemenge der Spalten im Referenzdatensatz übereinstimmt.

```
ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" = 1.0
```

Beispiel für dynamische Regeln

- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" > avg(last(10))`
- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" between min(last(10)) - 1 and max(last(10)) + 1`

RowCount

Prüft die Zeilenzahl eines Datensatzes anhand eines bestimmten Ausdrucks. Im Ausdruck können Sie mit Operatoren wie > und < die Anzahl der Zeilen oder einen Zeilenbereich angeben.

Syntax

```
RowCount <EXPRESSION>
```

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Numerische Überprüfung der Zeilenanzahl

Die folgende Beispielregel prüft, ob die Zeilenanzahl innerhalb eines bestimmten Bereichs liegt.

```
RowCount between 10 and 100
RowCount between 1 and 50 where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

```
RowCount > avg(last(10)) *0.8
```

RowCountMatch

Überprüft das Verhältnis der Zeilenanzahl des Primärdatensatzes und der Zeilenanzahl eines Referenzdatensatzes anhand des angegebenen Ausdrucks.

Syntax

```
RowCountMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS – Der Alias des Referenzdatensatzes, mit dem die Zeilenanzahl verglichen werden soll.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Überprüfung der Zeilenanzahl anhand eines Referenzdatensatzes

Die folgende Beispielregel überprüft, ob die Zeilenanzahl des Primärdatensatzes mindestens 90 % der Zeilenanzahl des Referenzdatensatzes beträgt.

```
RowCountMatch "reference" >= 0.9
```

StandardDeviation

Prüft die Standardabweichung aller Werte in einer Spalte anhand eines bestimmten Ausdrucks.

Syntax

```
StandardDeviation <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Example: Standardabweichung

Die folgende Beispielregel prüft, ob die Standardabweichung der Werte in einer Spalte mit dem Namen `colA` kleiner als ein bestimmter Wert ist.

```
StandardDeviation "Star_Rating" < 1.5
StandardDeviation "Salary" < 3500 where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `StandardDeviation "colA" > avg(last(10) + 0.1`
- `StandardDeviation "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Nullverhalten

Die `StandardDeviation` Regel ignoriert Zeilen mit `NULL` Werten bei der Berechnung der Standardabweichung. Zum Beispiel:

```
+---+-----------+-----------+
|id |units1     |units2     |
+---+-----------+-----------+
|100|0          |0          |
|101|null       |0          |
|102|20         |20         |
|103|null       |0          |
|104|40         |40         |
+---+-----------+-----------+
```

Bei der Standardabweichung der Spalte `units1` werden die Zeilen 101 und 103 nicht berücksichtigt und das Ergebnis ergibt 16,33. Die Standardabweichung für `units2` die Spalte ergibt 16.

Summe

Prüft die Summe aller Werte in einer Spalte anhand eines bestimmten Ausdrucks.

Syntax

```
Sum <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Summe

Die folgende Beispielregel prüft, ob die Summe aller Werte in einer Spalte einen bestimmten Schwellenwert überschreitet.

```
Sum "transaction_total" > 500000
Sum "Salary" < 55600 where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `Sum "ColA" > avg(last(10))`
- `Sum "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Null-Verhalten

Die `Sum` Regel ignoriert Zeilen mit `NULL` Werten bei der Summenberechnung. Zum Beispiel:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
```

```
+---+-----------+
```

Bei der Summe der Spalten `units` werden die Zeilen 101 und 103 nicht berücksichtigt und das Ergebnis ergibt (0 + 20 + 40) = 60.

SchemaMatch

Überprüft, ob das Schema des primären Datensatzes mit dem Schema eines Referenzdatensatzes übereinstimmt. Die Schemaprüfung erfolgt spaltenweise. Das Schema zweier Spalten stimmt überein, wenn die Namen identisch sind und die Typen identisch sind. Die Reihenfolge der Spalten spielt keine Rolle.

Syntax

```
SchemaMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS – Der Alias des Referenzdatensatzes, mit dem Schemas verglichen werden sollen.

  Unterstützte Spaltentypen: Byte, Dezimal, Doppelt, Gleitkommazahl, Ganzzahl, Lang, Kurz
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: SchemaMatch

Die folgende Beispielregel prüft, ob das Schema des Primärdatensatzes genau mit dem Schema eines Referenzdatensatzes übereinstimmt.

```
SchemaMatch "reference" = 1.0
```

Eindeutigkeit

Prüft den Prozentsatz eindeutiger Werte in einer Spalte anhand eines bestimmten Ausdrucks. Eindeutige Werte treten genau einmal auf.

Syntax

```
Uniqueness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

  Unterstützte Spaltentypen: Jeder Spaltentyp
- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel

Die folgende Beispielregel prüft, ob der Prozentsatz eindeutiger Werte in einer Spalte bestimmten numerischen Kriterien entspricht.

```
Uniqueness "email" = 1.0
Uniqueness "Customer_ID" != 1.0 where "Customer_ID < 10"
```

Die folgende Beispielregel überprüft mehrere Spalten.

```
Uniqueness "vendorid" "tpep_pickup_datetime" = 1
```

Beispiel für dynamische Regeln

- `Uniqueness "colA" between min(last(10)) and max(last(10))`
- `Uniqueness "colA" >= avg(last(10))`

UniqueValueRatio

Prüft das eindeutige Werteverhältnis einer Spalte anhand eines bestimmten Ausdrucks. Ein Einzelwertverhältnis ist der Bruchteil der Einzelwerte dividiert durch die Anzahl aller eindeutigen Werte in einer Spalte. Eindeutige Werte treten genau einmal auf, während unterschiedliche Werte mindestens einmal vorkommen.

Der Satz `[a, a, b]` enthält beispielsweise einen eindeutigen Wert (b) und zwei unterschiedliche Werte (a und b). Das eindeutige Wertverhältnis der Menge ist also ½ = 0,5.

Syntax

```
UniqueValueRatio <COL_NAME> <EXPRESSION>
```

- COL_NAME – Der Name der Spalte, anhand der Sie die Datenqualitätsregel auswerten möchten.

Unterstützte Spaltentypen: Jeder Spaltentyp

- AUSDRUCK – Ein Ausdruck, der für die Antwort des Regeltyps ausgeführt wird, um einen booleschen Wert zu erzeugen. Weitere Informationen finden Sie unter Ausdrücke.

Beispiel: Einzigartiges Werteverhältnis

In diesem Beispiel wird das Verhältnis der eindeutigen Werte einer Spalte zu einem Wertebereich geprüft.

```
UniqueValueRatio "test_score" between 0 and 0.5
UniqueValueRatio "Customer_ID" between 0 and 0.9 where "Customer_ID < 10"
```

Beispiel für dynamische Regeln

- `UniqueValueRatio "colA" > avg(last(10))`
- `UniqueValueRatio "colA" <= index(last(10),2) + std(last(5))`

DetectAnomalies

Erkennt Anomalien für eine bestimmte Datenqualitätsregel. Jede Ausführung einer DetectAnomalies Regel führt dazu, dass der ausgewertete Wert für die angegebene Regel gespeichert wird. Wenn genügend Daten gesammelt wurden, verwendet der Algorithmus zur Erkennung von Anomalien alle historischen Daten für diese bestimmte Regel und führt eine Anomalieerkennung durch. DetectAnomalies Die Regel schlägt fehl, wenn eine Anomalie erkannt wird. Weitere Informationen zu entdeckten Anomalien finden Sie in den Beobachtungen.

Syntax

```
    DetectAnomalies <RULE_NAME> <RULE_PARAMETERS>
```

`RULE_NAME` – Der Name der Regel, die ausgewertet und für die Anomalien erkannt werden sollen. Unterstützte Regeln:

- "RowCount"
- „Completeness“

- „Uniqueness“
- „Mean“
- „Sum“
- "StandardDeviation"
- „Entropy“
- "DistinctValuesCount"
- "UniqueValueRatio"
- "ColumnLength"
- "ColumnValues"
- "ColumnCorrelation"
- „Benutzerdefiniertes SQL“
- "ColumnCount"

`RULE_PARAMETERS` – Für die Ausführung einiger Regeln sind zusätzliche Parameter erforderlich. Die erforderlichen Parameter finden Sie in der jeweiligen Regeldokumentation.

Beispiel: Anomalien für RowCount

Wenn wir beispielsweise RowCount Anomalien erkennen möchten, geben wir RowCount als Regelnamen an.

```
DetectAnomalies "RowCount"
```

Beispiel: Anomalien für ColumnLength

Wenn wir beispielsweise ColumnLength Anomalien erkennen möchten, geben wir ColumnLength als Regelnamen und den Spaltennamen an.

```
DetectAnomalies "ColumnLength" "id"
```

FileFreshness

FileFreshness stellt sicher, dass Ihre Datendateien auf der Grundlage der von Ihnen angegebenen Bedingungen aktuell sind. Es verwendet den Zeitpunkt der letzten Änderung Ihrer Dateien, um sicherzustellen, dass die Datendateien oder der gesamte Ordner korrekt sind up-to-date.

Diese Regel erfasst zwei Messwerte:

- FileFreshness Einhaltung auf der Grundlage der von Ihnen eingerichteten Regel
- Die Anzahl der Dateien, die nach der Regel gescannt wurden

```
{"Dataset.*.FileFreshness.Compliance":1,"Dataset.*.FileCount":1}
```

Bei der Erkennung von Anomalien werden diese Messwerte nicht berücksichtigt.

Die Aktualität der Datei wird überprüft

Die folgende Regel stellt sicher, dass tickets.parquet in den letzten 24 Stunden erstellt wurde.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/tickets.parquet" >
 (now() - 24 hours)
```

Die Aktualität der Ordner wird überprüft

Die folgende Regel gilt, wenn alle Dateien im Ordner in den letzten 24 Stunden erstellt oder geändert wurden.

```
FileFreshness "s3://bucket/" >= (now() -1 days)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" >= (now() - 24 hours)
```

Überprüfung der Aktualität von Ordnern oder Dateien anhand eines Schwellenwerts

Die folgende Regel gilt als gültig, wenn 10% der Dateien im Ordner „Tickets“ in den letzten 10 Tagen erstellt oder geändert wurden.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" < (now() - 10 days)
 with threshold > 0.1
```

Überprüfung von Dateien oder Ordnern mit bestimmten Daten

Sie können die Aktualität von Dateien für bestimmte Tage überprüfen.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "2020-01-01"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "2023-01-01"
 and "2024-01-01"
```

Überprüfen Sie Dateien oder Ordner im Laufe der Zeit

Sie können FileFreshness damit sicherstellen, dass Dateien zu bestimmten Zeiten eingetroffen sind.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between now() and
 (now() - 45 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "9:30 AM" and
 "9:30 PM"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > (now() - 10 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > now()
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between (now() - 2
 hours) and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between (now() - 3
 days) and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "2001-02-07"
 and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "21:45""
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "2024-01-01"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "02:30" and
 "04:30"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "9:30 AM" and
 "22:15"
```

Wesentliche Überlegungen:

- FileFreshness kann Dateien anhand von Tagen, Stunden und Minuten auswerten
- Für Zeiten unterstützt es AM/PM und 24 Stunden
- Die Zeiten werden in UTC berechnet, sofern keine Überschreibung angegeben ist
- Datumsangaben werden in UTC zur Uhrzeit 00:00 berechnet

FileFreshness die zeitbasiert sind, funktionieren wie folgt:

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "21:45"
```

- Zunächst wird die Uhrzeit „21:45“ mit dem heutigen Datum im UTC-Format kombiniert, um ein Datum-/Uhrzeitfeld zu erstellen
- Als Nächstes wird das Datum und die Uhrzeit in eine von Ihnen angegebene Zeitzone konvertiert
- Schließlich wird die Regel ausgewertet

Optionale dateibasierte Regel-Tags:

Mithilfe von Tags können Sie das Regelverhalten steuern.

Letzte Dateien

Dieses Tag begrenzt die Anzahl der verarbeiteten Dateien, indem die neueste Datei zuerst angezeigt wird.

```
FileFreshness "s3://amzn-s3-demo-bucket/" between (now() - 100 minutes) and (now() + 10
 minutes) with recentFiles = 1
```

Zeitzone

Zulässige Zeitzonenüberschreibungen finden Sie unter Zulässige Zeitzonen für unterstützte Zeitzonen.

```
FileFreshness "s3://path/" > "21:45" with timeZone = "America/New_York"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "America/Chicago"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Europe/Paris"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Asia/Shanghai"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Australia/Darwin"
```

Direktes Ableiten von Dateinamen aus Datenrahmen

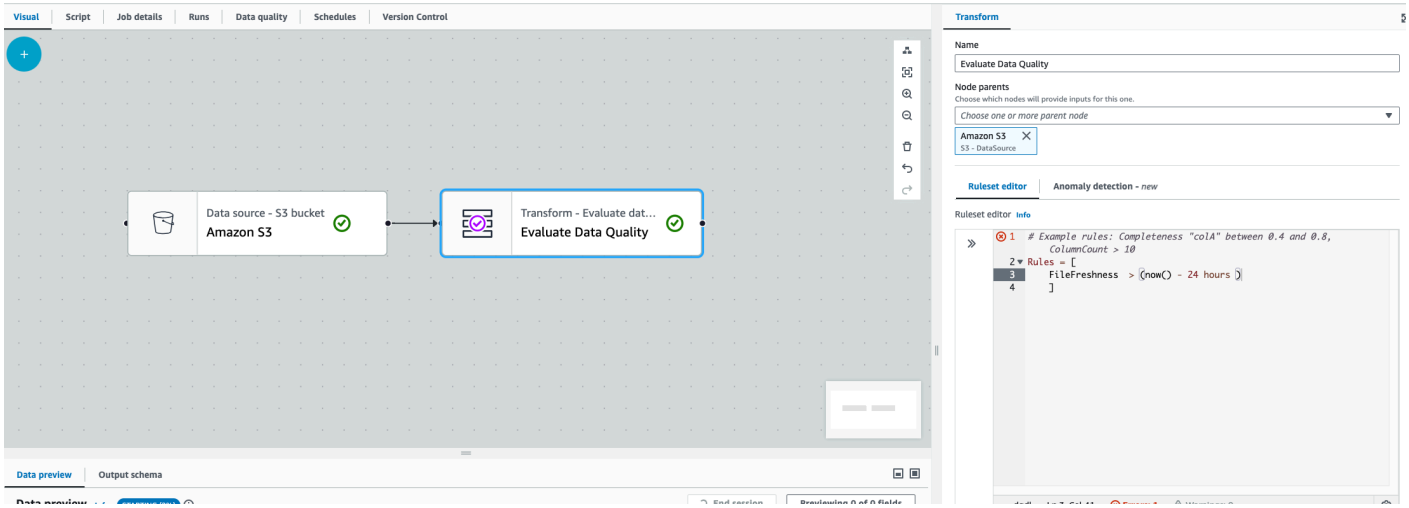
Sie müssen nicht immer einen Dateipfad angeben. Wenn Sie beispielsweise die Regel im AWS Glue-Datenkatalog erstellen, kann es schwierig sein, herauszufinden, welche Ordner die Katalogtabellen verwenden. AWS Glue Data Quality kann die spezifischen Ordner oder Dateien finden, die zum Auffüllen Ihres Datenrahmens verwendet wurden, und erkennt, ob sie aktuell sind.

> **Note**
>
> Diese Funktion funktioniert nur, wenn Dateien erfolgreich in das Feld oder eingelesen wurden. DynamicFrame DataFrame

```
FileFreshness > (now() - 24 hours)
```

Diese Regel findet den Ordnerpfad oder die Dateien, die zum Auffüllen des dynamischen Frames oder Datenrahmens verwendet werden. Dies funktioniert für Amazon S3 S3-Pfade oder Amazon S3-basierte AWS Glue Data Catalog-Tabellen. Es gibt ein paar Überlegungen:

1. In AWS Glue ETL müssen Sie die EvaluateDataQualityTransformation unmittelbar nach einer Amazon S3- oder AWS Glue Data Catalog-Transformation haben.



2. Diese Regel funktioniert nicht in AWS Glue Interactive Sessions.

Wenn Sie dies in beiden Fällen versuchen oder wenn AWS Glue die Dateien nicht finden kann, gibt AWS Glue den folgenden Fehler aus: `“Unable to parse file path from DataFrame”`

FileMatch

Die FileMatch Regel ermöglicht es Ihnen, Dateien mit anderen Dateien oder Prüfsummen zu vergleichen. Dies kann in einigen Szenarien nützlich sein:

1. Überprüfen von Dateien, die von externen Quellen empfangen wurden: Mit FileMatch dieser Methode können Sie sicherstellen, dass Sie die richtigen Dateien von externen Quellen erhalten haben, indem Sie sie mit Prüfsummen vergleichen. Dies hilft bei der Überprüfung der Integrität der Daten, die Sie aufnehmen.
2. Daten in zwei verschiedenen Ordnern vergleichen: FileMatch kann verwendet werden, um Dateien zwischen zwei Ordnern zu vergleichen.

Diese Regel erfasst eine Metrik: die Anzahl der Dateien, die mit der Regel gescannt wurden.

```
{"Dataset.*.FileCount":1}
```

Datei mit einer Prüfsumme validieren:

FileMatch akzeptiert eine Datei und festgelegte Prüfsummen, um sicherzustellen, dass mindestens eine Prüfsumme mit der Datei übereinstimmt.

```
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "MD5"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "SHA-1"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "SHA-256"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
```

Die folgenden Standardalgorithmen werden unterstützt:

- MD5
- SHA-1
- SHA-256

Wenn Sie keinen Algorithmus angeben, ist der Standard SHA-256.

Überprüfen Sie alle Dateien in einem Ordner mit einer Reihe von Prüfsummen:

```
FileMatch "s3://amzn-s3-demo-bucket /" in ["3ee0d8617ac041793154713e5ef8f319",
 "7e8617ac041793154713e5ef8f319"] with hashAlgorithm = "MD5"
FileMatch "s3://amzn-s3-demo-bucket /internal-folder/" in
 ["3ee0d8617ac041793154713e5ef8f319", "7e8617ac041793154713e5ef8f319"]
```

Vergleichen Sie Dateien in verschiedenen Ordnern

```
FileMatch "s3://original_bucket/" "s3://archive_bucket/"
FileMatch "s3://original_bucket/internal-folder/" "s3://original_bucket/other-folder/"
```

FileMatch überprüft den Inhalt der Dateien `original_bucket` und stellt sicher, dass sie mit dem Inhalt übereinstimmen`archive_bucket`. Die Regel schlägt fehl, wenn sie nicht genau übereinstimmen. Es kann auch den Inhalt interner Ordner oder einzelner Dateien überprüfen.

FileMatch kann auch einzelne Dateien miteinander vergleichen.

```
FileMatch "s3://amzn-s3-demo-bucket /file_old.json" "s3://amzn-s3-demo-bucket /
file_new.json"
```

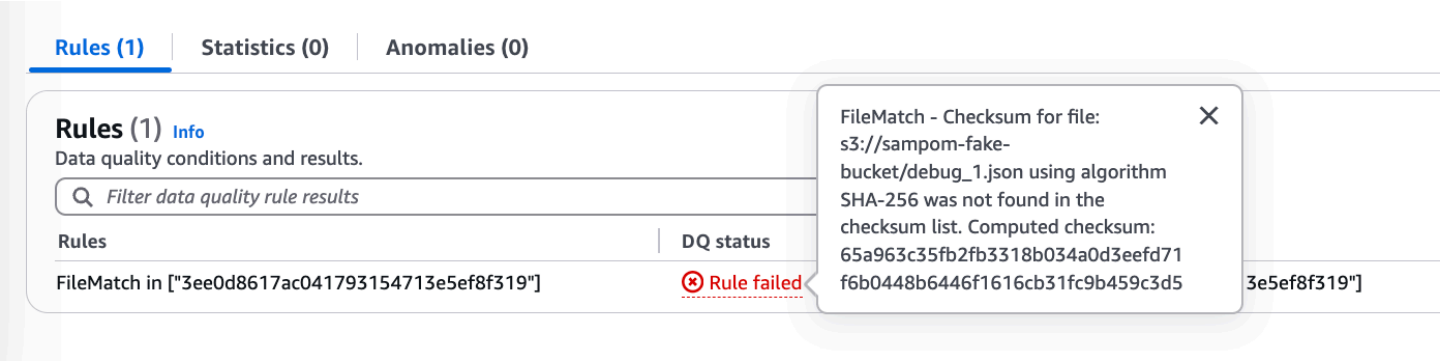
Dateinamen direkt aus Datenrahmen ableiten

Sie müssen nicht immer einen Dateipfad angeben. Wenn Sie beispielsweise die Regel im AWS Glue-Datenkatalog (unterstützt von Amazon S3) erstellen, kann es schwierig sein, herauszufinden, welche Ordner die Katalogtabellen verwenden. AWS Glue Data Quality kann die spezifischen Ordner oder Dateien finden, die zum Füllen Ihres Datenrahmens verwendet werden.

> **Note**
>
> Diese Funktion funktioniert nur, wenn Dateien erfolgreich in das Feld DynamicFrame oder DataFrame eingelesen wurden.

```
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "MD5"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "SHA-1"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "SHA-256"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"]
```

Wenn sich die angegebene Prüfsumme von der berechneten unterscheidet, FileMatch werden Sie auf den Unterschied hingewiesen.



Optionale dateibasierte Regel-Tags:

Mithilfe von Tags können Sie das Verhalten der Regel steuern.

Letzte Dateien

Dieses Tag begrenzt die Anzahl der verarbeiteten Dateien, indem die neueste Datei zuerst angezeigt wird.

```
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac04179sam4713e5ef8f319"]
 with recentFiles = 1
```

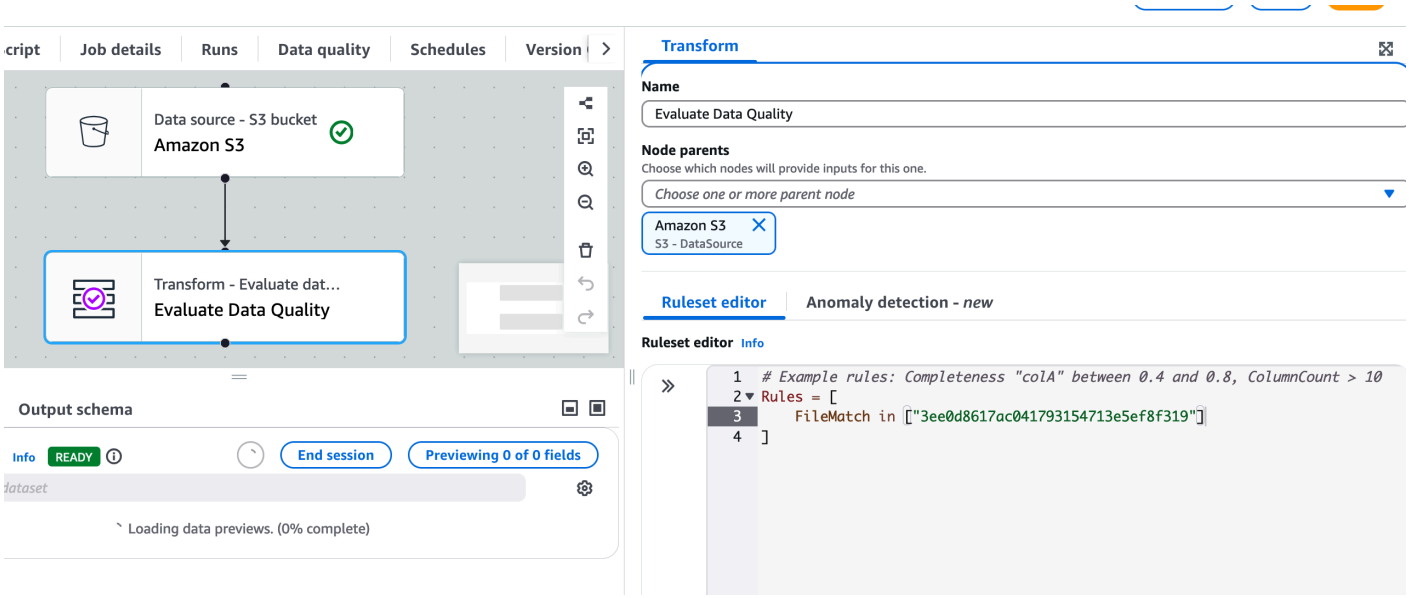
matchFileName

Dieses Tag stellt sicher, dass Dateien keine doppelten Namen haben. Das Standardverhalten ist falsch.

```
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac04179sam4713e5ef8f319"]
 with matchFileName = "true"
```

Es gibt ein paar Überlegungen:

1. In AWS Glue ETL müssen Sie die EvaluateDataQualityTransformation unmittelbar nach einer Amazon S3- oder AWS Glue Data Catalog-Transformation haben.



2. Diese Regel funktioniert nicht in AWS Glue Interactive Sessions.

FileUniqueness

Durch die Einzigartigkeit von Dateien können Sie sicherstellen, dass die Daten, die Sie von Ihren Datenproduzenten erhalten haben, keine doppelten Dateien enthalten.

Es sammelt die folgenden Datenstatistiken:

1. Die Anzahl der Dateien, die nach der Regel gescannt wurden
2. Das Eindeutigkeitsverhältnis der Dateien

```
Dataset.*.FileUniquenessRatio: 1.00, Dataset.*.FileCount: 8.00
```

Suchen Sie nach doppelten Dateien in einem Ordner:

```
FileUniqueness "s3://bucket/" > 0.5
FileUniqueness "s3://bucket/folder/" = 1
```

Ordnernamen direkt aus Datenrahmen ableiten, um Duplikate zu erkennen:

Sie müssen nicht immer einen Dateipfad angeben. Wenn Sie beispielsweise die Regel im AWS Glue-Datenkatalog erstellen, kann es schwierig sein, herauszufinden, welche Ordner die Katalogtabellen verwenden. AWS Glue Data Quality kann die spezifischen Ordner oder Dateien finden, die zum Füllen Ihres Datenrahmens verwendet werden.

> **Note**
>
> Bei der Verwendung von Inferenz können dateibasierte Regeln nur Dateien erkennen, die erfolgreich in das oder gelesen wurden. DynamicFrame DataFrame

```
FileUniqueness > 0.5
```

Optionale dateibasierte Regel-Tags:

Mithilfe von Tags können Sie das Verhalten der Regel steuern.

Letzte Dateien

Dieses Tag begrenzt die Anzahl der verarbeiteten Dateien, indem die neueste Datei zuerst angezeigt wird.

```
FileUniqueness "s3://amzn-s3-demo-bucket/" > 0.5 with recentFiles = 1
```

matchFileName

Dieses Tag stellt sicher, dass Dateien keine doppelten Namen haben. Das Standardverhalten ist falsch.

```
FileUniqueness "s3://amzn-s3-demo-bucket/" > 0.5 with matchFileName = "true"
```

Es gibt ein paar Überlegungen:

1. In AWS Glue ETL müssen Sie die EvaluateDataQualityTransformation unmittelbar nach einer Amazon S3- oder AWS Glue Data Catalog-Transformation haben.
2. Diese Regel funktioniert nicht in AWS Glue Interactive Sessions.

FileSize

Mit dem FileSize Regeltyp können Sie sicherstellen, dass Dateien bestimmte Dateigrößenkriterien erfüllen. Dies ist für folgende Anwendungsfälle nützlich:

1. Stellen Sie sicher, dass Hersteller keine leeren oder wesentlich kleineren Dateien zur Verarbeitung senden.
2. Stellen Sie sicher, dass Ihre Ziel-Buckets keine kleineren Dateien enthalten, was zu Leistungseinbußen führen kann.

FileSize sammelt die folgenden Messwerte:

1. Konformität: Gibt den Prozentsatz der Dateien zurück, die den von Ihnen festgelegten Regelschwellenwert erfüllen
2. Dateianzahl: Die Anzahl der Dateien, die nach der Regel gescannt wurden
3. Minimale Dateigröße in Byte
4. Maximale Dateigröße in Byte

```
Dataset.*.FileSize.Compliance: 1.00,
Dataset.*.FileCount: 8.00,
Dataset.*.MaximumFileSize: 327413121.00,
Dataset.*.MinimumFileSize: 204558920.00
```

Die Erkennung von Anomalien wird für diese Metriken nicht unterstützt.

Überprüfen Sie die Größe der Dateien

Diese Regel gilt, wenn file.dat größer als 2 MB ist.

```
FileSize "s3://amzn-s3-demo-bucket/file.dat" > 2 MB
```

Zu den unterstützten Einheiten gehören B (Byte), MB (Megabytes), GB (Gigabytes) und TB (Terabyte).

Überprüfen Sie die Größe der Dateien in Ordnern

```
FileSize "s3://bucket/" > 5 B
FileSize "s3://bucket/" < 2 GB
```

Diese Regel gilt als gültig, wenn 70% der Dateien auf s3://amzn-s3-demo-bucket zwischen 2 GB und 1 TB groß sind.

```
FileSize "s3://amzn-s3-demo-bucket/" between 2 GB and 1 TB  with threshold > 0.7
```

Dateinamen direkt aus Datenrahmen ableiten

Sie müssen nicht immer einen Dateipfad angeben. Wenn Sie beispielsweise die Regel im Datenkatalog erstellen, kann es schwierig sein, herauszufinden, welche Ordner die Katalogtabellen verwenden. AWS Glue Data Quality kann die spezifischen Ordner oder Dateien finden, die zum Füllen Ihres Datenrahmens verwendet werden.

> **Note**
>
> Diese Funktion funktioniert nur, wenn Dateien erfolgreich in das Feld DynamicFrame oder DataFrame eingelesen wurden.

```
FileSize < 10 MB with threshold > 0.7
```

Optionale dateibasierte Regel-Tags:

Mithilfe von Tags können Sie das Verhalten der Regel steuern.

Letzte Dateien

Dieses Tag begrenzt die Anzahl der verarbeiteten Dateien, indem die neueste Datei zuerst angezeigt wird.

```
FileSize "s3://amzn-s3-demo-bucket/" > 5 B with recentFiles = 1
```

matchFileName

Dieses Tag stellt sicher, dass Dateien keine doppelten Namen haben. Das Standardverhalten ist falsch.

```
FileSize "s3://amzn-s3-demo-bucket/" > 5 B with matchFileName = "true"
```

Es gibt ein paar Überlegungen:

1. In AWS Glue ETL muss Evaluate DataQuality Transform unmittelbar nach der Amazon S3- oder Data Catalog-Transformation installiert sein.
2. Diese Regel funktioniert nicht in AWS Glue Interactive Sessions.

Verwendung APIs zur Messung und Verwaltung der Datenqualität

In diesem Thema wird beschrieben, wie APIs die Datenqualität gemessen und verwaltet werden kann.

Inhalt

- Voraussetzungen
- Arbeiten mit AWS Glue-Datenqualitätsempfehlungen
- Mit AWS Glue Data Quality-Regelsätzen arbeiten
- Die Arbeit mit AWS Glue Data Quality läuft
- Arbeiten mit AWS Glue Data Quality-Ergebnissen

Voraussetzungen

- Stellen Sie sicher, dass Ihre boto3-Version auf dem neuesten Stand ist, sodass sie die neueste AWS Glue Data Quality API enthält.
- Stellen Sie sicher, dass Ihre AWS CLI-Version auf dem neuesten Stand ist, damit sie die neueste CLI enthält.

Wenn Sie einen AWS Glue-Job verwenden, um diese auszuführen APIs, können Sie die boto3-Bibliothek mit der folgenden Option auf die neueste Version aktualisieren:

```
–additional-python-modules boto3==<version>
```

Arbeiten mit AWS Glue-Datenqualitätsempfehlungen

Um einen AWS Glue Data Quality-Empfehlungslauf zu starten:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def start_data_quality_rule_recommendation_run(self, database_name, table_name,
 role_arn):
        """
        Starts a recommendation run that is used to generate rules when you don't know
 what rules to write. AWS Glue Data Quality
        analyzes the data and comes up with recommendations for a potential ruleset.
 You can then triage the ruleset
        and modify the generated ruleset to your liking.

        :param database_name: The name of the AWS Glue database which contains the
 dataset.
        :param table_name: The name of the AWS Glue table against which we want a
 recommendation
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
 Management (IAM) role that grants permission to let AWS Glue access the resources it
 needs.

        """
        try:
            response = self.client.start_data_quality_rule_recommendation_run(
                DataSource={
                    'GlueTable': {
                        'DatabaseName': database_name,
                        'TableName': table_name
                    }
                },
```

```
                Role=role_arn
            )
        except ClientError as err:
            logger.error(
                "Couldn't start data quality recommendation run %s. Here's why: %s:
 %s", name,
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response['RunId']
```

Bei einer Empfehlungsausführung können Sie Ihre `pushDownPredicates` oder `catalogPartitionPredicates` verwenden, um die Leistung zu verbessern und Empfehlungen nur für bestimmte Partitionen Ihrer Katalogquellen auszuführen.

```
client.start_data_quality_rule_recommendation_run(
            DataSource={
                'GlueTable': {
                    'DatabaseName': database_name,
                    'TableName': table_name,
                    'AdditionalOptions': {
                        'pushDownPredicate': "year=2022"
                    }
                }
            },
            Role=role_arn,
            NumberOfWorkers=2,
            CreatedRulesetName='<rule_set_name>'
  )
```

Um die Ergebnisse einer AWS Glue Data Quality-Empfehlung zu erhalten, gehen Sie wie folgt vor:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_rule_recommendation_run(self, run_id):
        """
```

```
        Gets the specified recommendation run that was used to generate rules.

        :param run_id: The id of the data quality recommendation run

        """
        try:
            response =
 self.client.get_data_quality_rule_recommendation_run(RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get data quality recommendation run %. Here's why: %s: %s",
 run_id,
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Aus dem obigen Antwortobjekt können Sie das Objekt extrahieren RuleSet , das im Lauf empfohlen wurde, um es in weiteren Schritten zu verwenden:

```
print(response['RecommendedRuleset'])

Rules = [
    RowCount between 2000 and 8000,
    IsComplete "col1",
    IsComplete "col2",
    StandardDeviation "col3" between 58138330.8 and 64258155.09,
    ColumnValues "col4" between 1000042965 and 1214474826,
    IsComplete "col5"
]
```

So erhalten Sie eine Liste aller Ihrer Empfehlungsausführungen, die gefiltert und aufgelistet werden können:

```
response = client.list_data_quality_rule_recommendation_runs(
    Filter={
        'DataSource': {
            'GlueTable': {
                'DatabaseName': '<database_name>',
                'TableName': '<table_name>'
            }
        }
```

```
)
```

So stornieren AWS Sie bestehende Glue Data Quality-Empfehlungsaufgaben:

```
response = client.cancel_data_quality_rule_recommendation_run(
    RunId='dqrun-d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)
```

Mit AWS Glue Data Quality-Regelsätzen arbeiten

So erstellen Sie einen AWS Glue Data Quality-Regelsatz:

```
response = client.create_data_quality_ruleset(
    Name='<ruleset_name>',
    Ruleset='Rules = [IsComplete "col1", IsPrimaryKey "col2", RowCount between 2000 and
 8000]',
    TargetTable={
        'TableName': '<table_name>',
        'DatabaseName': '<database_name>'
    }
)
```

So erhalten Sie einen Datenqualitätsregelsatz:

```
response = client.get_data_quality_ruleset(
    Name='<ruleset_name>'
)
print(response)
```

Mit dieser API können Sie anschließend den Regelsatz extrahieren:

```
print(response['Ruleset'])
```

So listen Sie alle Datenqualitätsregelsätze für eine Tabelle auf:

```
response = client.list_data_quality_rulesets()
```

Mit der Filterbedingung innerhalb der API können Sie alle Regelsätze filtern, die einer bestimmten Datenbank oder Tabelle zugeordnet sind:

```
response = client.list_data_quality_rulesets(
```

```
    Filter={
        'TargetTable': {
            'TableName': '<table_name>',
            'DatabaseName': '<database_name>'
        }
    },
)
```

So aktualisieren Sie einen Datenqualitätsregelsatz:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def update_data_quality_ruleset(self, ruleset_name, ruleset_string):
        """
        Update an AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to update
        :param ruleset_string: The DQDL ruleset string to update the ruleset with

        """
        try:
            response = self.client.update_data_quality_ruleset(
                Name=ruleset_name,
                Ruleset=ruleset_string
            )
        except ClientError as err:
            logger.error(
                "Couldn't update the AWS Glue Data Quality ruleset. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

So aktualisieren Sie einen Datenqualitätsregelsatz:

```
class GlueWrapper:
```

```
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def delete_data_quality_ruleset(self, ruleset_name):
        """
        Delete a AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to delete

        """
        try:
            response = self.client.delete_data_quality_ruleset(
                Name=ruleset_name
            )
        except ClientError as err:
            logger.error(
                "Couldn't delete the AWS Glue Data Quality ruleset. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Die Arbeit mit AWS Glue Data Quality läuft

Um einen AWS Glue Data Quality-Lauf zu starten:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def start_data_quality_ruleset_evaluation_run(self, database_name, table_name,
 role_name, ruleset_list):
        """
        Start an AWS Glue Data Quality evaluation run
```

```
        :param database_name: The name of the AWS Glue database which contains the
    dataset.
        :param table_name: The name of the AWS Glue table against which we want to
    evaluate.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
    Management (IAM) role that grants permission to let AWS Glue access the resources it
    needs.
        :param ruleset_list: The list of AWS Glue Data Quality ruleset names to
    evaluate.

        """
        try:
            response = client.start_data_quality_ruleset_evaluation_run(
                DataSource={
                    'GlueTable': {
                        'DatabaseName': database_name,
                        'TableName': table_name
                    }
                },
                Role=role_name,
                RulesetNames=ruleset_list
            )
        except ClientError as err:
            logger.error(
                "Couldn't start the AWS Glue Data Quality Run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response['RunId']
```

Denken Sie daran, dass Sie einen `pushDownPredicate`- oder `catalogPartitionPredicate`-Parameter übergeben können, um festzulegen, dass Ihre Datenqualitätsausführung nur auf eine bestimmte Partition innerhalb Ihrer Katalogtabelle abzielt. Zum Beispiel:

```
response = client.start_data_quality_ruleset_evaluation_run(
    DataSource={
        'GlueTable': {
            'DatabaseName': '<database_name>',
            'TableName': '<table_name>',
            'AdditionalOptions': {
                'pushDownPredicate': 'year=2023'
            }
```

```
        }
    },
    Role='<role_name>',
    NumberOfWorkers=5,
    Timeout=123,
    AdditionalRunOptions={
        'CloudWatchMetricsEnabled': False
    },
    RulesetNames=[
        '<ruleset_name>',
    ]
)
```

Sie können auch konfigurieren, wie zusammengesetzte Regeln in Ihrem Regelsatz ausgewertet werden, entweder auf der ROW- oder COLUMN-Ebene. Weitere Informationen zur Funktionsweise von Verbundregeln finden Sie in der Dokumentation unter So funktionieren Verbundregeln.

Beispiel dafür, wie Sie die Bewertungsmethode für zusammengesetzte Regeln in Ihrer Anfrage festlegen können:

```
response = client.start_data_quality_ruleset_evaluation_run(
    DataSource={
        'GlueTable': {
            'DatabaseName': '<database_name>',
            'TableName': '<table_name>',
            'AdditionalOptions': {
                'pushDownPredicate': 'year=2023'
            }
        }
    },
    Role='<role_name>',
    NumberOfWorkers=5,
    Timeout=123,
    AdditionalRunOptions={
        'CompositeRuleEvaluationMethod':ROW
    },
    RulesetNames=[
        '<ruleset_name>',
    ]
)
```

Um Informationen über einen AWS Glue Data Quality-Lauf zu erhalten:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_ruleset_evaluation_run(self, run_id):
        """
        Get details about an AWS Glue Data Quality Run

        :param run_id: The AWS Glue Data Quality run ID to look up

        """
        try:
            response = self.client.get_data_quality_ruleset_evaluation_run(
                RunId=run_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't look up the AWS Glue Data Quality run ID. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Um die Ergebnisse eines AWS Glue Data Quality-Laufs zu erhalten:

Für einen bestimmten AWS Glue Data Quality-Lauf können Sie die Ergebnisse der Auswertung des Laufs mit der folgenden Methode extrahieren:

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)
```

```
print(response['RuleResults'])
```

Um alle Ihre AWS Glue Data Quality-Läufe aufzulisten:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def list_data_quality_ruleset_evaluation_runs(self, database_name, table_name):
        """
        Lists all the AWS Glue Data Quality runs against a given table

        :param database_name: The name of the database where the data quality runs
        :param table_name: The name of the table against which the data quality runs
 were created

        """
        try:
            response = self.client.list_data_quality_ruleset_evaluation_runs(
                Filter={
                    'DataSource': {
                        'GlueTable': {
                            'DatabaseName': database_name,
                            'TableName': table_name
                        }
                    }
                }
            )
        except ClientError as err:
            logger.error(
                "Couldn't list the AWS Glue Quality runs. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Sie können die Filterklausel so ändern, dass nur Ergebnisse zwischen bestimmten Zeiten oder bei der Ausführung für bestimmte Tabellen angezeigt werden.

Um einen laufenden AWS Glue Data Quality-Lauf zu beenden:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def cancel_data_quality_ruleset_evaluation_run(self, result_id):
        """
        Cancels a given AWS Glue Data Quality run

        :param result_id: The result id of a AWS Glue Data Quality run to cancel

        """
        try:
            response = self.client.cancel_data_quality_ruleset_evaluation_run(
                ResultId=result_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't cancel the AWS Glue Data Quality run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Arbeiten mit AWS Glue Data Quality-Ergebnissen

So erhalten Sie die Ergebnisse Ihres AWS Glue Data Quality-Laufs:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_result(self, result_id):
        """
```

```
        Outputs the result of an AWS Glue Data Quality Result

        :param result_id: The result id of an AWS Glue Data Quality run

        """
        try:
            response = self.client.get_data_quality_result(
                ResultId=result_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't get the AWS Glue Data Quality result. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

So zeigen Sie die Statistiken an, die für ein bestimmtes Datenqualitätsergebnis gesammelt wurden:

```
import boto3
from botocore.exceptions import ClientError
import logging

logger = logging.getLogger(__name__)
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_profile_for_data_quality_result(self, result_id):
        """
        Outputs the statistic profile for a AWS Glue Data Quality Result

        :param result_id: The result id of a AWS Glue Data Quality run

        """
        try:
            response = self.glue_client.get_data_quality_result(
                ResultId=result_id
            )
```

```
            # the profile contains all statistics gathered for the result
            profile_id = response['ProfileId']
            profile = self.glue_client.list_data_quality_statistics(
                ProfileId = profile_id
            )
            return profile
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality profile. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

So zeigen Sie die Zeitreihen für eine Statistik an, die in mehreren Datenqualitätsläufen erfasst wurde:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_statistics_for_data_quality_result(self, profile_id):
        """
        Outputs an array of datapoints for each statistic in the input result.

        :param result_id: The profile id of a AWS Glue Data Quality run

        """
        try:
            profile = self.glue_client.list_data_quality_statistics(
                ProfileId = profile_id
            )
            statistics = [self.glue_client.list_data_quality_statistics(
                StatisticId = s['StatisticId']
            ) for s in profile['Statistics']]
            return statistics
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality statistics. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

So zeigen Sie das Modell zur Erkennung von Anomalien für eine bestimmte Statistik an:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_model_training_result_for_statistic(self, statistic_id, profile_id):
        """
        Outputs the details (bounds) of anomaly detection training for the given
 statistic at the given profile.

        :param statistic_id the model's statistic (the timeseries it is tracking)
        :param profile_id the profile associated with the model (a point in the
 timeseries)

        """
        try:
            model = self.glue_client.get_data_quality_model_result(
                ProfileId = profile_id, StatisticId = statistic_id
            )
            return model
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality model results. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Um einen Datenpunkt von der Basis für die Anomalieerkennung seines Statistikmodells auszuschließen:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def apply_exclusions_to_statistic(self, statistic_id, profile_ids):
```

```
        """
        Annotate some points along a given statistic timeseries.

        This example excludes the provided values; INCLUDE can also be used to undo
 this action.

        :param statistic_id the statistic timeseries to annotate
        :param profile_id the profiles we want to exclude (points in the timeseries)

        """

        try:
            response = self.glue_client.batch_put_data_quality_statistic_annotation(
                    InclusionAnnotations = [
                        {'ProfileId': prof_id,
                        'StatisticId': statistic_id,
                        'InclusionAnnotation': 'EXCLUDE'} for prof_id in profile_ids
                    ]
            )
            return response['FailedInclusionAnnotations']
        except ClientError as err:
            logger.error(
                "Couldn't store Data Quality annotations. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

So zeigen Sie den Status des Trainings eines Modells zur Erkennung von Anomalien für eine bestimmte Statistik an:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_model_training_status_for_statistic(self, statistic_id, profile_id):
        """
        Outputs the status of anomaly detection training for the given statistic at the
 given profile.

        :param statistic_id the model's statistic (the timeseries it is tracking)
```

```
        :param profile_id the profile associated with the model (a point in the
 timeseries)

        """
        try:
            model = self.glue_client.get_data_quality_model(
                ProfileId = profile_id, StatisticId = statistic_id
            )
            return model
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality statistics. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Um alle Ergebnisse eines bestimmten Datenqualitätslaufs von den Ausgangswerten für die Anomalieerkennung auszuschließen, gehen Sie wie folgt vor:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def apply_exclusions_to_profile(self, profile_id):
        """
        Exclude datapoints produced by a run across statistic timeseries.

        This example excludes the provided values; INCLUDE can also be used to undo
 this action.

        :param profile_id the profiles we want to exclude (points in the timeseries)

        """
        try:
            response = self.glue_client.put_data_quality_profile_annotation(
                    ProfileId = profile_id,
                    InclusionAnnotation = "EXCLUDE"
            )
            return response
        except ClientError as err:
```

```
            logger.error(
                "Couldn't store Data Quality annotations. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

So rufen Sie die Ergebnisse eines bestimmten Datenqualitätslaufs ab und zeigen die Ergebnisse an:

Mit einer AWS Glue-Datenqualität können Sie die extrahieren`runID`, `resultID` um dann die tatsächlichen Ergebnisse zu erhalten, wie unten gezeigt:

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='dqrun-abca77ee126abe1378c1da1ae0750d7dxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)

print(resp['RuleResults'])
```

Einrichten von Warnmeldungen, Bereitstellungen und Planung

In diesem Thema wird beschrieben, wie Sie Benachrichtigungen, Bereitstellungen und Zeitpläne für AWS Glue Data Quality einrichten.

Inhalt

- Benachrichtigungen und Benachrichtigungen in der EventBridge Amazon-Integration einrichten
  - Zusätzliche Konfigurationsoptionen für das Ereignismuster
  - Formatierung von Benachrichtigungen als E-Mails
- Richten Sie Warnmeldungen und Benachrichtigungen in der CloudWatch Integration ein
- Abfragen von Datenqualitätsergebnissen zum Erstellen von Dashboards
- Bereitstellung von Datenqualitätsregeln mit AWS CloudFormation
- Planung von Datenqualitätsregeln

Benachrichtigungen und Benachrichtigungen in der EventBridge Amazon-Integration einrichten

AWS Glue Data Quality unterstützt die Veröffentlichung von EventBridge Ereignissen, die nach Abschluss eines Evaluierungslaufs für den Data Quality-Regelsatz ausgegeben werden. Damit können Sie ganz einfach Warnmeldungen einrichten, wenn Datenqualitätsregeln fehlschlagen.

Hier sehen Sie ein Beispiel für ein Ereignis, wenn Sie Datenqualitätsregeln im Datenkatalog bewerten. Mit diesen Informationen können Sie die Daten überprüfen, die bei Amazon zur Verfügung gestellt EventBridge werden. Sie können zusätzliche API-Aufrufe ausgeben, um weitere Details zu erhalten. Rufen Sie beispielsweise die `get_data_quality_result`-API mit der Ergebnis-ID auf, um die Details einer bestimmten Ausführung abzurufen.

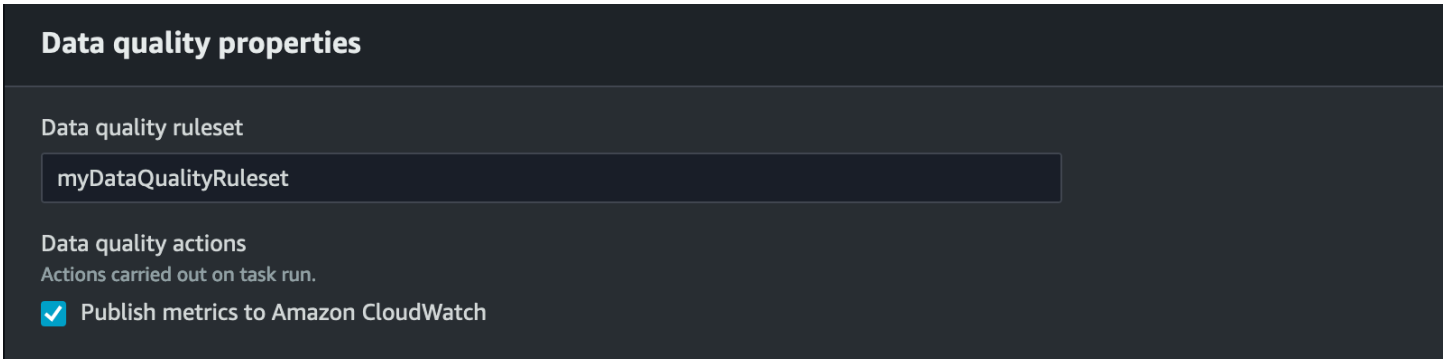
```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_DATA_CATALOG",
                    "runId":"dqrun-12334567890",
                    "databaseName": "db-123",
                    "tableName": "table-123",
                    "catalogId": "123456789012"
                    },
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00,
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

Hier ist ein Beispielereignis, das veröffentlicht wird, wenn Sie Regelsätze für die Datenqualität in AWS Glue ETL- oder AWS Glue Studio-Notebooks auswerten.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_JOB",
                    "jobId": "jr-12334567890",
                    "jobName": "dq-eval-job-1234",
                    "evaluationContext": "",
                    }
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

Damit die Datenqualitätsbewertung sowohl im Datenkatalog als auch in ETL-Jobs ausgeführt wird, muss die Amazon CloudWatch Option Metriken veröffentlichen in, die standardmäßig aktiviert ist, aktiviert bleiben, damit die EventBridge Veröffentlichung funktioniert.

EventBridge Benachrichtigungen einrichten



Um die ausgesendeten Ereignisse zu empfangen und Ziele zu definieren, müssen Sie EventBridge Amazon-Regeln konfigurieren. So erstellen Sie Regeln:

1. Öffnen Sie die EventBridge Amazon-Konsole.
2. Wählen Sie die Option Regeln im Abschnitt Busse in der Navigationsleiste aus.
3. Wählen Sie Create Rule (Regel erstellen) aus.
4. Gehen Sie unter Regeldetails definieren wie folgt vor:
   a. Geben Sie als Name `myDQRule` ein.
   b. Geben Sie die Beschreibung ein (optional).
   c. Wählen Sie für den Event Bus Ihren Event Bus aus. Wenn Sie noch keinen haben, belassen Sie es als Standard.
   d. Wählen Sie als Regeltyp Regel mit einem Ereignis-Muster und wählen Sie dann Weiter.
5. Unter Ereignis-Muster entwickeln:
   a. Wählen Sie als Quelle für die Veranstaltung AWS Veranstaltungen oder EventBridge Partnerveranstaltungen aus.
   b. Überspringen Sie den Abschnitt mit den Beispielereignissen.
   c. Wählen Sie als Erstellungsmethode die Option Musterformular verwenden aus.
   d. Für das Ereignismuster:
      i. Wählen Sie AWS -Services für die Ereignisquelle aus.
      ii. Wählen Sie Glue Data Quality für den AWS Service aus.
      iii. Wählen Sie für den Ereignistyp die Option Verfügbare Ergebnisse der Datenqualitätsbewertung aus.
      iv. Wählen Sie FEHLGESCHLAGEN für bestimmte(n) Status(en) aus. Sie sehen dann ein Ereignismuster, ähnlich dem folgenden:

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "state": ["FAILED"]
  }
}
```

v. Weitere Konfigurationsoptionen finden Sie unter Zusätzliche Konfigurationsoptionen für das Ereignismuster.

6. Unter Ziel(e) auswählen:
   a. Wählen Sie unter Zieltypen den AWS -Service aus.
   b. Verwenden Sie die Dropdownliste Ziel auswählen, um den gewünschten AWS Dienst auszuwählen, zu dem Sie eine Verbindung herstellen möchten (SNS, Lambda, SQS usw.), und wählen Sie dann Weiter aus.
7. Klicken Sie unter Tag(s) konfigurieren auf Neue Tags hinzufügen, um optionale Tags hinzuzufügen, und wählen Sie dann Weiter aus.
8. Es wird eine Übersichtsseite aller Auswahlen angezeigt. Wählen Sie unten die Option Regel erstellen.

Zusätzliche Konfigurationsoptionen für das Ereignismuster

Zusätzlich zum Filtern Ihres Ereignisses nach Erfolg oder Misserfolg möchten Sie möglicherweise Ereignisse weiter nach verschiedenen Parametern filtern.

Gehen Sie dazu zum Abschnitt Ereignismuster und wählen Sie Muster bearbeiten, um zusätzliche Parameter anzugeben. Beachten Sie, dass bei Feldern im Ereignismuster die Groß- und Kleinschreibung beachtet wird. Im Folgenden finden Sie Beispiele für die Konfiguration des Ereignismusters.

Um Ereignisse aus einer bestimmten Tabelle zu erfassen und bestimmte Regelsätze auszuwerten, verwenden Sie diesen Mustertyp:

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
```

```
    "context": {
      "contextType": ["GLUE_DATA_CATALOG"],
      "databaseName": "db-123",
       "tableName": "table-123",
    },
    "rulesetNames": ["ruleset1", "ruleset2"]
    "state": ["FAILED"]
  }
}
```

Um Ereignisse von bestimmten Aufträgen in der ETL-Erfahrung zu erfassen, verwenden Sie diesen Mustertyp:

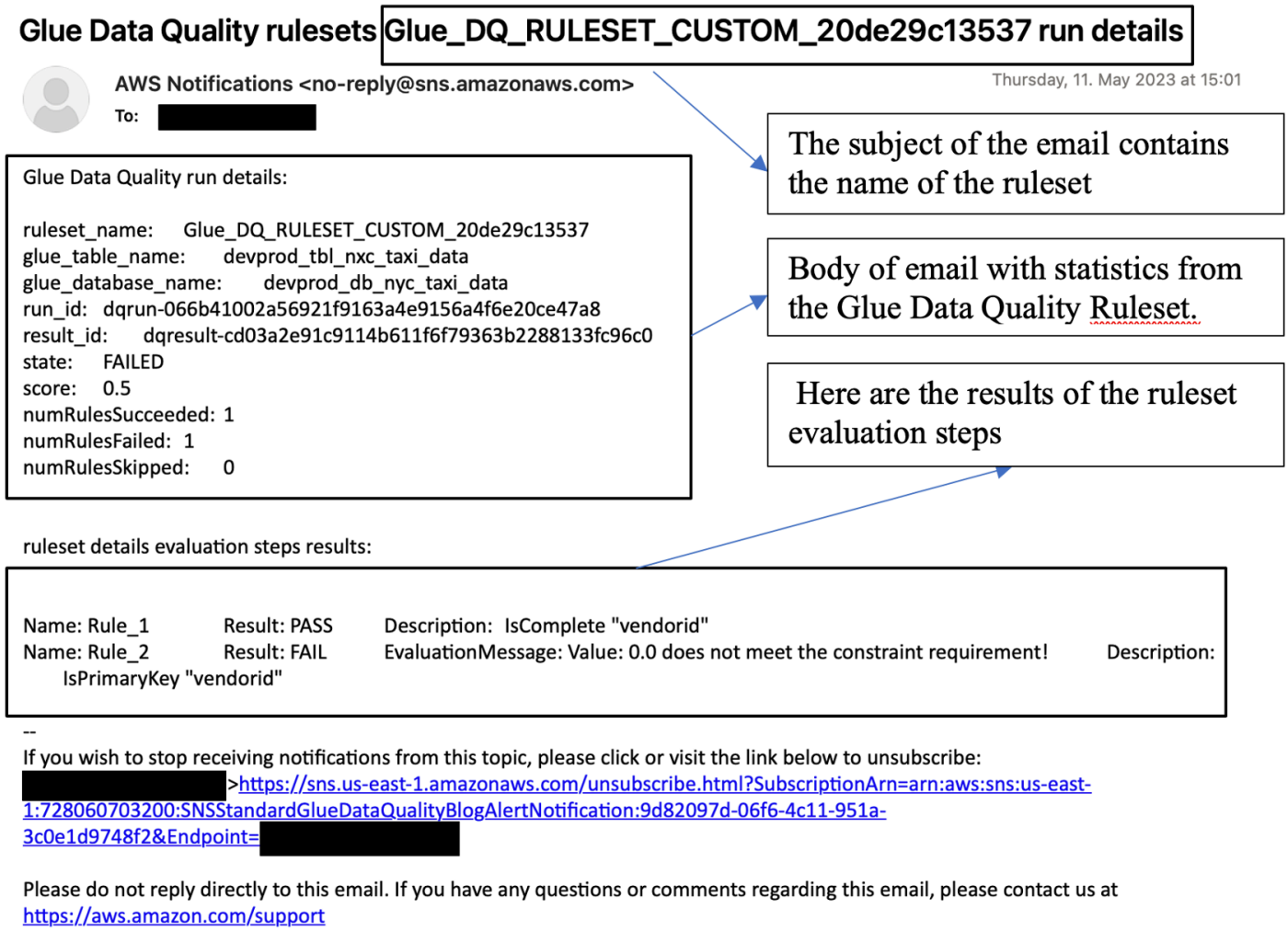
```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "context": {
      "contextType": ["GLUE_JOB"],
      "jobName": ["dq_evaluation_job1", "dq_evaluation_job2"]
    },
    "state": ["FAILED"]
  }
}
```

So erfassen Sie Ereignisse mit einem Wert unter einem bestimmten Schwellenwert (z. B. 70 %):

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "score": [{
      "numeric": ["<=", 0.7]
    }]
  }
}
```

Formatierung von Benachrichtigungen als E-Mails

Manchmal müssen Sie eine gut formatierte E-Mail-Benachrichtigung an Ihre Unternehmensteams senden. Sie können Amazon EventBridge und AWS Lambda verwenden, um dies zu erreichen.



Mit dem folgenden Beispielcode können Sie Ihre Datenqualitätsbenachrichtigungen formatieren, um E-Mails zu generieren.

```
import boto3
import json
from datetime import datetime

sns_client = boto3.client('sns')
glue_client = boto3.client('glue')

sns_topic_arn = 'arn:aws:sns:<region-code>:<account-id>:<sns-topic-name>'



def lambda_handler(event, context):
    log_metadata = {}
```

```
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
```

```
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info



    sns_client.publish(
        TopicArn=sns_topic_arn,
        Message=message_text,
        Subject=subject_text
    )

    return {
```

```
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Richten Sie Warnmeldungen und Benachrichtigungen in der CloudWatch Integration ein

Wir empfehlen, Datenqualitätswarnungen über Amazon einzurichten EventBridge, da Amazon eine einmalige Einrichtung EventBridge benötigt, um Kunden zu benachrichtigen. Einige Kunden bevorzugen jedoch Amazon CloudWatch aufgrund der Vertrautheit. Für solche Kunden bieten wir die Integration mit Amazon an CloudWatch.

Jede AWS Glue Data Quality-Bewertung gibt pro Datenqualitätslauf ein Paar von Metriken mit den Namen `glue.data.quality.rules.passed` (was die Anzahl der Regeln angibt, die bestanden haben) und `glue.data.quality.rules.failed` (die Anzahl der fehlgeschlagenen Regeln) aus. Mit dieser ausgegebenen Metrik können Sie Warnmeldungen erstellen, um Benutzer zu benachrichtigen, wenn eine bestimmte Ausführung der Datenqualität unter einen Schwellenwert fällt. Führen Sie die folgenden Schritte aus, um mit dem Einrichten einer Warnmeldung zu beginnen, der eine E-Mail über eine Amazon-SNS-Benachrichtigung sendet:

Führen Sie die folgenden Schritte aus, um mit dem Einrichten einer Warnmeldung zu beginnen, der eine E-Mail über eine Amazon-SNS-Benachrichtigung sendet:

1. Öffnen Sie die CloudWatch Amazon-Konsole.
2. Wählen Sie Alle Metriken unter Metriken aus. Unter Benutzerdefinierte Namespaces wird ein zusätzlicher Namespace mit dem Titel Glue Data Quality angezeigt.

   > **Note**
   >
   > Wenn AWS Sie einen Glue Data Quality-Lauf starten, stellen Sie sicher, dass das CloudWatch Kontrollkästchen Metriken auf Amazon veröffentlichen aktiviert ist. Andernfalls werden die Metriken für diesen bestimmten Lauf nicht auf Amazon veröffentlicht CloudWatch.

   Unter dem `Glue Data Quality`-Namespace können Sie die ausgegebenen Metriken pro Tabelle und pro Regelsatz sehen. Für die Zwecke dieses Themas verwenden wir die

`glue.data.quality.rules.failed`-Regel und den Alarm, wenn dieser Wert 1 überschreitet (was bedeutet, dass wir benachrichtigt werden möchten, wenn wir eine Anzahl fehlgeschlagener Regelauswertungen größer als 1 sehen).

3. Um den Alarm zu erstellen, wählen Sie Alle Alarme unter Alarme aus.
4. Wählen Sie Create alarm (Alarm erstellen) aus.
5. Wählen Sie Select metric (Metrik auswählen) aus.
6. Wählen Sie die `glue.data.quality.rules.failed`-Metrik aus, die der von Ihnen erstellten Tabelle entspricht, und wählen Sie dann Metrik auswählen aus.
7. Gehen Sie auf der Registerkarte Metrik und Bedingungen angeben im Abschnitt Metriken wie folgt vor:
   a. Wählen Sie für Statistic (Statistik) Sum (Summe) aus.
   b. Wählen Sie als Zeitraum die Option 1 Minute aus.
8. Im Abschnitt Bedingungen:
   a. Wählen Sie für Threshold type (Schwellenwerttyp) die Option Static (Statisch) aus.
   b. Wählen Sie für Immer wenn glue.data.quality.rules.failed ist... die Option Größer/Gleich aus.
   c. Für als... , geben Sie 1 als Schwellenwert ein.

   Diese Auswahl bedeutet, dass wir einen Alarm auslösen, wenn die `glue.data.quality.rules.failed`-Metrik einen Wert größer oder gleich 1 ausgibt. Wenn jedoch keine Daten vorhanden sind, behandeln wir diese als akzeptabel.
9. Wählen Sie Weiter.
10. Gehen Sie unter Aktionen konfigurieren wie folgt vor:
    a. Wählen Sie für den Abschnitt Alarmstatus-Auslöser die Option Im Alarm aus.
    b. Wählen Sie im Abschnitt Benachrichtigung an folgendes SNS-Thema senden die Option Neues Thema erstellen zum Senden einer Benachrichtigung über ein neues SNS-Thema aus.
    c. Geben Sie für E-Mail-Endpunkte, die die Benachrichtigung erhalten, Ihre E-Mail-Adresse ein. Klicken Sie dann auf Thema erstellen.
    d. Wählen Sie Weiter.
11. Geben Sie für Alarm-Name `myFirstDQAlarm` ein und klicken Sie auf Weiter.
12. Es wird eine Übersichtsseite aller Auswahlen angezeigt. Wählen Sie unten Alarm erstellen aus.

Sie können jetzt im Amazon-Alarm-Dashboard sehen, wie der CloudWatch Alarm erstellt wird.

Abfragen von Datenqualitätsergebnissen zum Erstellen von Dashboards

Möglicherweise möchten Sie ein Dashboard erstellen, um Ihre Datenqualitätsergebnisse anzuzeigen. Es gibt zwei Möglichkeiten dafür:

Richten Sie Amazon EventBridge mit dem folgenden Code ein, um die Daten in Amazon S3 zu schreiben:

```
import boto3
import json
from datetime import datetime


s3_client = boto3.client('s3')
glue_client = boto3.client('glue')


s3_bucket = 's3-bucket-name'

def write_logs(log_metadata):
    try:
        filename = datetime.now().strftime("%m%d%Y%H%M%S") + ".json"
        key_opts = {
            'year': datetime.now().year,
            'month': "{:02d}".format(datetime.now().month),
            'day': "{:02d}".format(datetime.now().day),
            'filename': filename
        }
        s3key = "gluedataqualitylogs/year={year}/month={month}/day={day}/
{filename}".format(**key_opts)
        s3_client.put_object(Bucket=s3_bucket, Key=s3key,
 Body=json.dumps(log_metadata))
    except Exception as e:
        print(f'Error writing logs to S3: {e}')


def lambda_handler(event, context):
    log_metadata = {}
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
```

```
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
```

```
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info

    write_logs(log_metadata)


    return {
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Nachdem Sie in Amazon S3 geschrieben haben, können Sie AWS Glue-Crawler verwenden, um sich bei Athena zu registrieren und die Tabellen abzufragen.

Konfigurieren Sie während einer Datenqualitätsbewertung einen Amazon-S3-Speicherort:

Wenn Sie Datenqualitätsaufgaben im AWS Glue Data Catalog oder AWS Glue ETL ausführen, können Sie einen Amazon S3-Speicherort angeben, um die Datenqualitätsergebnisse in Amazon S3 zu schreiben. Mit der folgenden Syntax können Sie eine Tabelle erstellen, indem Sie auf das Ziel verweisen, um die Ergebnisse der Datenqualität zu lesen.

Beachten Sie, dass Sie die `CREATE EXTERNAL TABLE`- und `MSCK REPAIR TABLE`-Abfragen separat ausführen müssen.

```
CREATE EXTERNAL TABLE <my_table_name>(
    catalogid string,
    databasename string,
    tablename string,
    dqrunid string,
    evaluationstartedon timestamp,
    evaluationcompletedon timestamp,
    rule string,
    outcome string,
    failurereason string,
    evaluatedmetrics string)
PARTITIONED BY (
    `year` string,
    `month` string,
    `day` string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (

 'paths'='catalogId,databaseName,dqRunId,evaluatedMetrics,evaluationCompletedOn,evaluationStart
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://glue-s3-dq-bucket-us-east-2-results/'
TBLPROPERTIES (
    'classification'='json',
    'compressionType'='none',
    'typeOfData'='file');
```

```
MSCK REPAIR TABLE <my_table_name>;
```

Sobald Sie die obige Tabelle erstellt haben, können Sie mit Amazon Athena analytische Abfragen ausführen.

Bereitstellung von Datenqualitätsregeln mit AWS CloudFormation

Sie können AWS CloudFormation es verwenden, um Datenqualitätsregeln zu erstellen. Weitere Informationen finden Sie unter AWS CloudFormation AWS Glue.

Planung von Datenqualitätsregeln

Sie können Datenqualitätsregeln mit den folgenden Methoden planen:

- Datenqualitätsregeln aus dem Datenkatalog planen: Benutzer ohne Programmierkenntnisse können diese Option verwenden, um ihre Datenqualitätsscans einfach zu planen. AWS Glue Data Quality erstellt den Zeitplan in Amazon EventBridge. So planen Sie Datenqualitätsregeln:
  - Navigieren Sie zum Regelsatz und klicken Sie auf Ausführen.
  - Wählen Sie im Feld Ausführungshäufigkeit den gewünschten Zeitplan aus und geben Sie einen Aufgabennamen an. Dieser Aufgabenname ist der Name Ihres Zeitplans in EventBridge.
- Verwenden Sie Amazon EventBridge und AWS Step Functions, um Bewertungen und Empfehlungen für Datenqualitätsregeln zu orchestrieren.

Datenverschlüsselung im Ruhezustand für AWS Glue Data Quality

AWS Glue Data Qualitybietet standardmäßig Verschlüsselung, um vertrauliche Kundendaten im Speicher mithilfe AWS eigener Verschlüsselungsschlüssel zu schützen.

AWS eigene Schlüssel

AWS Glue Data Quality verwendet diese Schlüssel, um die Datenqualitätsressourcen der Kunden automatisch zu verschlüsseln. Sie können AWS eigene Schlüssel nicht einsehen, verwalten oder verwenden oder deren Verwendung überprüfen. Sie müssen jedoch keine Maßnahmen ergreifen oder Programme ändern, um die Schlüssel zu schützen, mit denen Ihre Daten verschlüsselt werden. Weitere Informationen finden Sie AWS im AWS KMS Entwicklerhandbuch unter Eigene Schlüssel.

Die standardmäßige Verschlüsselung von Daten im Ruhezustand trägt dazu bei, den betrieblichen Aufwand und die Komplexität zu reduzieren, die mit dem Schutz vertraulicher Daten verbunden sind. Gleichzeitig können Sie damit sichere Anwendungen erstellen, die strenge Verschlüsselungsvorschriften und gesetzliche Auflagen erfüllen.

Sie können diese Verschlüsselungsebene zwar nicht deaktivieren oder einen anderen Verschlüsselungstyp auswählen, Sie können jedoch eine zweite Verschlüsselungsebene über den

vorhandenen AWS eigenen Verschlüsselungsschlüsseln hinzufügen, indem Sie bei der Erstellung Ihrer Data Quality-Ressourcen einen vom Kunden verwalteten Schlüssel auswählen.

Kundenverwaltete Schlüssel

Vom Kunden verwaltete Schlüssel: AWS Glue Data Quality unterstützt die Verwendung eines symmetrischen, vom Kunden verwalteten Schlüssels, den Sie erstellen, besitzen und verwalten. Dadurch wird eine zweite Verschlüsselungsebene gegenüber der bestehenden AWS eigenen Verschlüsselung hinzugefügt. Da Sie die volle Kontrolle über diese Verschlüsselungsebene haben, können Sie Aufgaben wie die folgenden ausführen:

- Festlegung und Pflege wichtiger Richtlinien
- Einrichtung und Pflege von IAM-Richtlinien
- Aktivieren und Deaktivieren wichtiger Richtlinien
- Kryptographisches Material mit rotierendem Schlüssel
- Hinzufügen von Tags
- Erstellen von Schlüsselaliasen
- Schlüssel für das Löschen von Schlüsseln planen

Weitere Informationen finden Sie unter Vom Kunden verwaltete Schlüssel im AWS KMS Entwicklerhandbuch.

Die folgende Tabelle fasst zusammen, wie AWS Glue Data Quality verschiedene Data Quality-Ressourcen verschlüsselt.

Datentyp	AWS Verschlüsselung mit eigenem Schlüssel	Vom Kunden verwaltete Schlüsselverschlüsselung
Regelsatz für die Datenqualität DQDL-Regelsatzzeichenfolge, auf die der persistente DQ-Regelsatz verweist. Diese persistenten Regelsätze werden derzeit nur im AWS Glue Data Catalog verwendet.	Aktiviert	Aktiviert

Datentyp	AWS Verschlüsselung mit eigenem Schlüssel	Vom Kunden verwaltete Schlüsselverschlüsselung
Ergebnisse der Datenqualitätsregel/des Analyzers Ergebnisartefakte, die den Status „Bestehen/Nicht bestanden“ jeder Regel in einem Regelsatz sowie Messwerte, die sowohl von Regeln als auch von Analysatoren erfasst wurden, enthalten.	Aktiviert	Aktiviert
Beobachtungen Beobachtungen werden generiert, wenn eine Anomalie in den Daten festgestellt wird. Es enthält Informationen über die erwartete Ober- und Untergrenze sowie eine vorgeschlagene Regel, die auf diesen Grenzen basiert. Falls sie generiert wurden, werden sie zusammen mit den Datenqualitätsergebnissen angezeigt.	Aktiviert	Aktiviert

Datentyp	AWS Verschlüsselung mit eigenem Schlüssel	Vom Kunden verwaltete Schlüsselverschlüsselung
Statistiken Enthält Informationen zu Metriken, die nach der Auswertung der Daten anhand eines Regelsatzes erfasst wurden, z. B. den Wert der Metrik (z. B. RowCount Vollständigkeit), Spaltenna men und andere Metadaten.	Aktiviert	Aktiviert
Statistische Modelle zur Erkennung von Anomalien Die statistischen Modelle enthalten die Zeitreihen der Ober- und Untergrenzen für eine bestimmte Kennzahl, die auf der Grundlage früherer Bewertungen der Kundendat en generiert wurden.	Aktiviert	Aktiviert

> **Note**
>
> AWS Data Quality ermöglicht automatisch die Verschlüsselung im Ruhezustand mithilfe AWS eigener Schlüssel, um personenbezogene Daten kostenlos zu schützen. Für die Verwendung eines vom Kunden verwalteten Schlüssels fallen jedoch AWS KMS Gebühren an. Weitere Informationen über die Preise finden Sie unter AWS KMS – Preise.
> Weitere Informationen zu finden AWS KMS Sie unter AWS KMS.

Erstellen Sie einen vom Kunden verwalteten Schlüssel

Sie können einen symmetrischen, vom Kunden verwalteten Schlüssel erstellen, indem Sie den AWS Management Console, oder den AWS KMS APIs verwenden.

Einen symmetrischen kundenverwalteten Schlüssel erstellen:

- Folgen Sie den Schritten zum Erstellen symmetrischer AWS KMS Verschlüsselungsschlüssel im AWS Key Management Service Entwicklerhandbuch.

Schlüsselrichtlinie

Schlüsselrichtlinien steuern den Zugriff auf den vom Kunden verwalteten Schlüssel. Jeder vom Kunden verwaltete Schlüssel muss über genau eine Schlüsselrichtlinie verfügen, die aussagt, wer den Schlüssel wie verwenden kann. Wenn Sie Ihren vom Kunden verwalteten Schlüssel erstellen, können Sie eine Schlüsselrichtlinie angeben. Weitere Informationen finden Sie im AWS Key Management Service Entwicklerhandbuch unter Wichtige Richtlinien in AWS KMS Schlüsseln.

Um Ihren vom Kunden verwalteten Schlüssel mit Ihren Data Quality-Ressourcen zu verwenden, müssen die folgenden API-Operationen in der Schlüsselrichtlinie zulässig sein:

- `kms:Decrypt`— Entschlüsselt Chiffretext, der mit einem Schlüssel verschlüsselt wurde AWS KMS `GenerateDataKeyWithoutPlaintext`
- `kms:DescribeKey`— Stellt die vom Kunden verwalteten Schlüsselinformationen bereit, damit Amazon Location den Schlüssel validieren kann.
- `kms:GenerateDataKeyWithoutPlaintext`— Gibt einen eindeutigen symmetrischen Datenschlüssel zur Verwendung außerhalb von AWS KMS zurück. Dieser Vorgang gibt einen Datenschlüssel zurück, der unter einem von Ihnen angegebenen symmetrischen Verschlüsselungs-KMS-Schlüssel verschlüsselt ist. Die Byte im Schlüssel sind zufällig; sie beziehen sich nicht auf den Anrufer oder den KMS-Schlüssel. Wird verwendet, um die Anzahl der KMS-Anrufe zu reduzieren, die der Kunde tätigen muss.
- `kms:ReEncrypt*`— Entschlüsselt den Chiffretext und verschlüsselt ihn dann vollständig im Inneren erneut. AWS KMS Sie können diesen Vorgang verwenden, um den KMS-Schlüssel zu ändern, mit dem Daten verschlüsselt werden, z. B. wenn Sie einen KMS-Schlüssel manuell rotieren oder den KMS-Schlüssel ändern, der einen Chiffretext schützt. Sie können ihn auch verwenden, um Chiffretext unter demselben KMS-Schlüssel erneut zu verschlüsseln, z. B. um den Verschlüsselungskontext eines Chiffretextes zu ändern.

Im Folgenden finden Sie Beispiele für Richtlinienerklärungen, die Sie für Amazon Location hinzufügen können:

```
"Statement" : [
    {
        "Sid" : "Allow access to principals authorized to use AWS Glue Data Quality",
        "Effect" : "Allow",
        "Principal" : {
            "AWS": "arn:aws:iam::<account_id>:role/ExampleRole"
        },
        "Action" : [
            "kms:Decrypt",
            "kms:DescribeKey",
            "kms:GenerateDataKeyWithoutPlaintext",
            "kms:ReEncrypt*"
        ],
        "Resource" : "*",
        "Condition" : {
            "StringEquals" : {
                "kms:ViaService" : "glue.amazonaws.com",
                "kms:CallerAccount" : "111122223333"
            }
        },
    {
        "Sid": "Allow access for key administrators",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:root"
          },
        "Action" : [
            "kms:*"
         ],
        "Resource": "arn:aws:kms:region:111122223333:key/key_ID"
    },
    {
        "Sid" : "Allow read-only access to key metadata to the account",
        "Effect" : "Allow",
        "Principal" : {
            "AWS" : "arn:aws:iam::111122223333:root"
        },
        "Action" : [
            "kms:Describe*",
            "kms:Get*",
```

```
            "kms:List*",
        ],
        "Resource" : "*"
    }
]
```

Hinweise zur Verwendung von KMS-Schlüsseln in AWS Glue Data Quality

AWS Glue Data Quality unterstützt keine wichtigen Übergänge. Das heißt, wenn Sie Ihre Datenqualitätsressourcen mit Schlüssel A verschlüsseln und sich entscheiden, zu Schlüssel B zu wechseln, werden wir die mit Schlüssel A verschlüsselten Daten nicht erneut verschlüsseln, um Schlüssel B zu verwenden. Sie können immer noch zu Schlüssel B wechseln, müssen aber weiterhin Zugriff auf Schlüssel A haben, um auf Daten zugreifen zu können, die zuvor mit Schlüssel A verschlüsselt wurden.

Weitere Informationen zum Angeben von Berechtigungen in einer Richtlinie finden Sie im Entwicklerhandbuch unter Berechtigungen für AWS Dienste in wichtigen Richtlinien. AWS Key Management Service

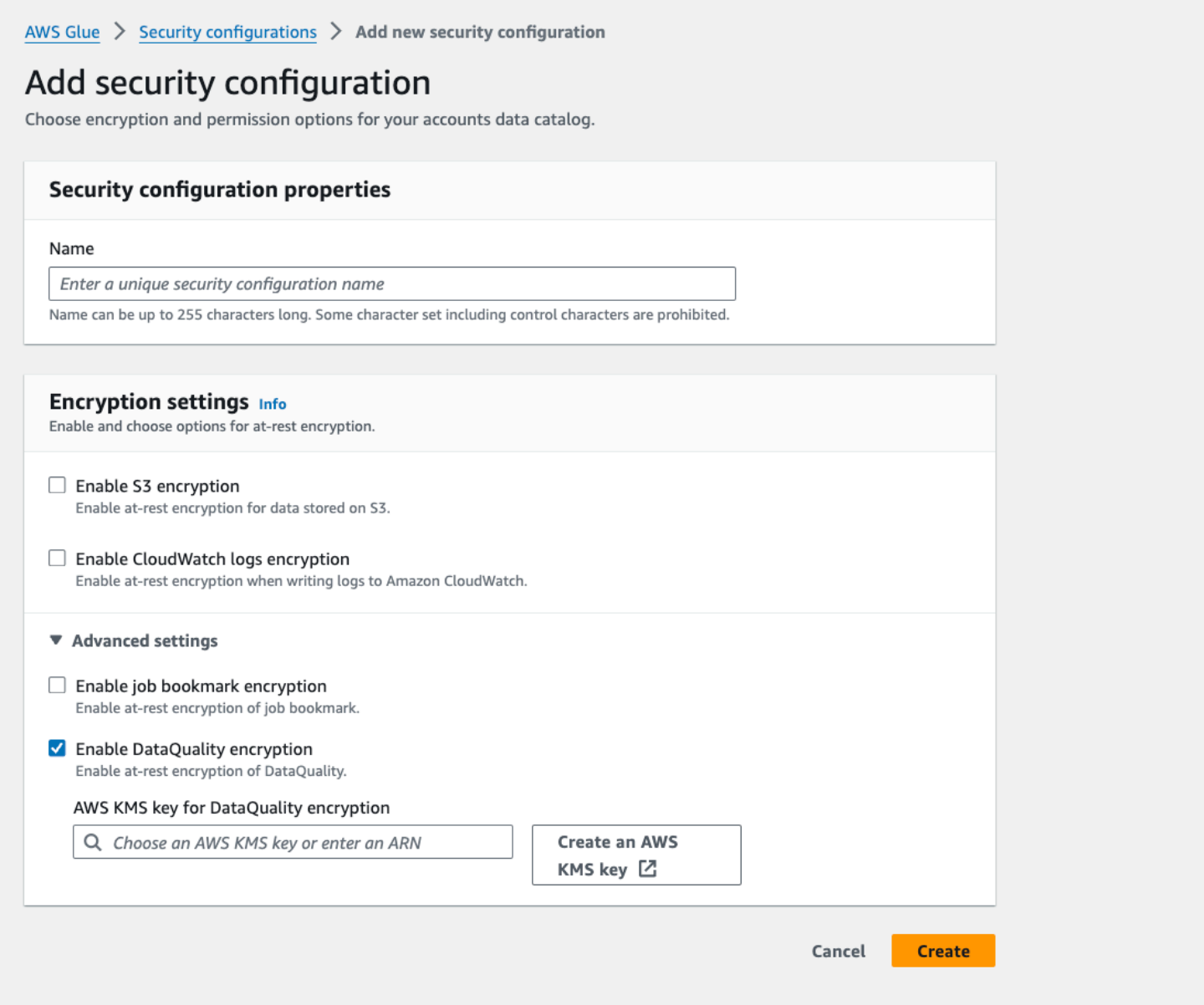
Weitere Informationen zur Problembehandlung beim Schlüsselzugriff finden Sie unter Problembehandlung beim Schlüsselzugriff im AWS Key Management Service Entwicklerhandbuch.

Eine Sicherheitskonfiguration erstellen

In AWS Glue enthält die Ressource Sicherheitskonfigurationen Eigenschaften, die beim Schreiben verschlüsselter Daten benötigt werden.

So verschlüsseln Sie Ihre Datenqualitätsressourcen:

1. Wählen Sie in den Verschlüsselungseinstellungen unter Erweiterte Einstellungen die Option Datenqualitätsverschlüsselung aktivieren
2. Wählen Sie Ihren KMS-Schlüssel aus oder wählen Sie AWS KMS Schlüssel erstellen



AWS Glue Data Quality-Verschlüsselungskontext

Ein Verschlüsselungskontext ist ein optionaler Satz von Schlüssel-Wert-Paaren, die zusätzliche kontextbezogene Informationen zu den Daten enthalten.

AWS KMS verwendet den Verschlüsselungskontext als zusätzliche authentifizierte Daten, um die authentifizierte Verschlüsselung zu unterstützen. Wenn Sie einen Verschlüsselungskontext in eine Anforderung zum Verschlüsseln von Daten einbeziehen, wird der Verschlüsselungskontext AWS KMS an die verschlüsselten Daten gebunden. Zur Entschlüsselung von Daten müssen Sie denselben Verschlüsselungskontext in der Anfrage übergeben.

AWS Beispiel für einen Glue Data Quality-Verschlüsselungskontext

```
"encryptionContext": {
    "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "hierarchy-version": "1",
    "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-security-
configuration-name",
    "create-time": "2024-06-07T13:47:23:000861Z",
    "tablename": "AwsGlueMlEncryptionKeyStore",
    "type": "beacon:ACTIVE"
}
```

Verwenden des Verschlüsselungskontexts für die Überwachung

Wenn Sie einen symmetrischen, vom Kunden verwalteten Schlüssel verwenden, um Ihre Tracker- oder Geofence-Erfassung zu verschlüsseln, können Sie den Verschlüsselungskontext auch in Prüfaufzeichnungen und Protokollen verwenden, um zu ermitteln, wie der vom Kunden verwaltete Schlüssel verwendet wird. Der Verschlüsselungskontext erscheint auch in Protokollen, die von oder generiert wurden. AWS CloudTrail Amazon CloudWatch Logs

Überwachung Ihrer Verschlüsselungsschlüssel für AWS Glue Data Quality

Wenn Sie einen vom AWS KMS Kunden verwalteten Schlüssel mit Ihren AWS Glue Data Quality-Ressourcen verwenden, können Sie AWS CloudTrail oder verwenden, Amazon CloudWatch Logs um Anfragen zu verfolgen, an die AWS Glue Data Quality sendet AWS KMS.

Die folgenden Beispiele sind AWS CloudTrail Ereignisse für `GenerateDataKeyWithoutPlainText` und `Decrypt` zur Überwachung von KMS-Vorgängen, die von AWS Glue Data Quality aufgerufen werden, um auf Daten zuzugreifen, die mit Ihrem vom Kunden verwalteten Schlüssel verschlüsselt wurden.

Decrypt

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
```

```
        "accountId": "111122223333",
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-02T20:03:10Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:ACTIVE",
            "version": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
```

```
}
```

GenerateDataKeyWithoutPlaintext

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
        "accountId": "111122223333",
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-02T20:03:10Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

ReEncyrpt

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
        "accountId": "111122223333",
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-17T21:34:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ReEncrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "destinationEncryptionContext": {
             "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:ACTIVE"
            "version": "branch:version:12345678-SAMPLE"
        },
        "destinationKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "sourceAAD": "1234567890-SAMPLE+Z+lqoYOHj7VtWxJLrvh+biUFbliYDAQkobM=",
```

```
        "sourceKeyId": "arn:aws:kms:ap-southeast-2:585824196334:key/17ca05ca-a8c1-40d7-
b7fd-30abb569a53a",
        "destinationEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "sourceEncryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        },
        "destinationAAD": "1234567890-SAMPLE",
        "sourceEncryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Weitere Informationen

Die folgenden Ressourcen enthalten weitere Informationen zur Datenverschlüsselung im Ruhezustand:

- Weitere Informationen zu AWS Key Management Service grundlegenden Konzepten finden Sie im AWS Key Management Service Entwicklerhandbuch.
- Weitere Informationen zu bewährten Sicherheitsmethoden AWS Key Management Service finden Sie im AWS Key Management Service Entwicklerhandbuch.

Behebung von AWS Glue-Datenqualitätsfehlern

Wenn Sie bei AWS Glue Data Quality auf Fehler stoßen, verwenden Sie die folgenden Lösungen, um die Ursache der Probleme zu finden und sie zu beheben.

Inhalt

- Fehler: Fehlendes AWS Glue Data Quality-Modul
- Fehler: unzureichende AWS Lake Formation Formation-Berechtigungen
- Fehler: Regelsätze sind nicht eindeutig benannt
- Fehler: Tabellen mit Sonderzeichen
- Fehler: Überlauffehler bei einem großen Regelsatz
- Fehler: der allgemeine Regelstatus ist fehlgeschlagen
- AnalysisException: Das Vorhandensein der Standarddatenbank konnte nicht überprüft werden
- Fehlermeldung: Die bereitgestellte Schlüsselzuordnung ist für die angegebenen Datenrahmen nicht geeignet
- Ausnahme in der Benutzerklasse: java.lang. RuntimeException : Daten konnten nicht abgerufen werden. Schauen Sie in den Logs CloudWatch nach, um weitere Informationen zu erhalten
- STARTFEHLER: Fehler beim Herunterladen von S3 für den Bucket
- InvalidInputException (Status: 400): DataQuality Regeln können nicht analysiert werden
- Fehler: Eventbridge löst basierend auf dem von mir eingerichteten Zeitplan keine Glue-DQ-Aufträge aus
- CustomSQL-Fehler
- Dynamische Regeln

- Ausnahme in der Benutzerklasse: org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException
- UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing-Fehler: Es wurden keine Regeln oder Analysatoren bereitgestellt., keine praktikable Alternative bei der Eingabe

Fehler: Fehlendes AWS Glue Data Quality-Modul

Fehlermeldung: Kein Modul mit dem Namen „awsgluedq“.

Lösung: Dieser Fehler tritt auf, wenn Sie AWS Glue Data Quality in einer nicht unterstützten Version ausführen. AWS Glue Data Quality wird nur in Glue Version 3.0 und höher unterstützt.

Fehler: unzureichende AWS Lake Formation Formation-Berechtigungen

Fehlermeldung: Ausnahme in User Class:`com.amazonaws.services.glue.model.AccessDeniedException`: Unzureichende Lake Formation Formation-Berechtigungen für impact_sdg_involvement (Service: AWS Glue; Statuscode: 400; Fehlercode:; Anforderungs-ID: 465ae693-b7ba-4df0-a4e4-6b17xxxxxxxx AccessDeniedException; Proxy: null).

Lösung: Sie müssen ausreichende Berechtigungen in AWS Lake Formation bereitstellen.

Fehler: Regelsätze sind nicht eindeutig benannt

Fehlermeldung: Ausnahme in der Benutzerklasse:... services.glue.model. AlreadyExistsException: Ein anderer Regelsatz mit demselben Namen ist bereits vorhanden.

Lösung: Regelsätze sind global und müssen eindeutig sein.

Fehler: Tabellen mit Sonderzeichen

Fehlermeldung: Ausnahme in der Benutzerklasse: org.apache.spark.sql. AnalysisException: kann „C“ bei gegebenen Eingabespalten nicht auflösen: [primary.data_end_time, primary.data_start_time, primary.end_time, primary.last_updated, primary.message, primary.process_date, primary.rowhash, primary.run_by, primary.run_id, primary.start_time, primary.status]; Zeile 1 Pos 44;.

Lösung: Derzeit besteht die Einschränkung, dass AWS Glue Data Quality nicht für Tabellen ausgeführt werden kann, die Sonderzeichen wie „.“ enthalten.

Fehler: Überlauffehler bei einem großen Regelsatz

Fehlermeldung: Ausnahme in der Benutzerklasse: java.lang. StackOverflowError.

Lösung: Wenn Sie über einen großen Regelsatz mit mehr als 2 000 Regeln verfügen, kann dieses Problem auftreten. Teilen Sie Ihre Regeln in mehrere Regelsätze auf.

Fehler: der allgemeine Regelstatus ist fehlgeschlagen

Fehlerbedingung: Mein Regelsatz ist erfolgreich, aber mein Gesamtregelstatus ist fehlgeschlagen.

Lösung: Dieser Fehler ist höchstwahrscheinlich darauf zurückzuführen, dass Sie CloudWatch während der Veröffentlichung die Option ausgewählt haben, Metriken auf Amazon zu veröffentlichen. Wenn sich Ihr Datensatz in einer VPC befindet, erlaubt Ihre VPC AWS Glue möglicherweise nicht, Metriken auf Amazon zu veröffentlichen. CloudWatch In diesem Fall müssen Sie >einen Endpunkt einrichten, damit Ihre VPC auf Amazon zugreifen kann. CloudWatch

AnalysisException: Das Vorhandensein der Standarddatenbank konnte nicht überprüft werden

Fehlerbedingung AnalysisException: Das Vorhandensein der Standarddatenbank konnte nicht überprüft werden: com.amazonaws.services.glue.model. AccessDeniedException: Standardmäßig sind keine ausreichenden Lake Formation Formation-Berechtigungen vorhanden (Service: AWS Glue; Statuscode: 400; Fehlercode: AccessDeniedException; Anforderungs-ID: XXXXXXXX-XXXX-XXXX-XXXX -XXXXXXXXXXXX; Proxy: null)

Auflösung: In AWS Glue Integration des Auftragskatalogs, AWS Glue versucht immer zu überprüfen, ob die Standarddatenbank existiert oder nicht AWS Glue `GetDatabase API`. Wenn die `DESCRIBE` Lake Formation Formation-Berechtigung nicht erteilt wird oder die `GetDatabase IAM` Berechtigung erteilt wird, schlägt der Job fehl, wenn überprüft wird, ob die Standarddatenbank vorhanden ist.

Um dies zu lösen:

1. Fügen Sie in Lake Formation die `DESCRIBE`-Berechtigung für die Standarddatenbank hinzu.
2. Konfigurieren Sie die IAM-Rolle, die dem zugewiesen ist AWS Glue Job als Datenbankentwickler bei Lake Formation. Dadurch wird automatisch eine Standarddatenbank erstellt und die erforderlichen Lake-Formation-Berechtigungen für die Rolle gewährt.
3. Option `--enable-data-catalog` deaktivieren. (Es wird als Verwenden angezeigt Data Catalog als Hive-Metastore in AWS Glue Studio).

Wenn Sie Spark SQL nicht benötigen Data Catalog Integration in den Job, Sie können sie deaktivieren.

Fehlermeldung: Die bereitgestellte Schlüsselzuordnung ist für die angegebenen Datenrahmen nicht geeignet

Fehlerbedingung: Die bereitgestellte Schlüsselzuordnung ist für die angegebenen Datenrahmen nicht geeignet.

Lösung: Sie verwenden den DataSetMatchRegeltyp und die Join-Schlüssel haben Duplikate. Ihre Verknüpfungsschlüssel müssen eindeutig sein und dürfen nicht NULL sein. In Fällen, in denen Sie keine eindeutigen Join-Schlüssel haben können, sollten Sie die Verwendung anderer Regeltypen in Betracht ziehen, z. B. AggregateMatchfür den Abgleich bei Übersichtsdaten.

Ausnahme in der Benutzerklasse: java.lang. RuntimeException : Daten konnten nicht abgerufen werden. Schauen Sie in den Logs CloudWatch nach, um weitere Informationen zu erhalten

Fehlerzustand: Ausnahme in der Benutzerklasse: java.lang. RuntimeException : Daten konnten nicht abgerufen werden. Schauen Sie in den Logs CloudWatch nach, um weitere Informationen zu erhalten.

Lösung: Dies passiert, wenn Sie DQ-Regeln für eine Amazon S3-basierte Tabelle erstellen, die mit Amazon RDS oder verglichen wird. Amazon Redshift In diesen Fällen AWS Glue kann die Verbindung nicht laden. Versuchen Sie stattdessen, eine DQ-Regel für den Amazon Redshift oder Amazon RDS-Datensatz einzurichten. Dies ist ein bekannter Fehler.

STARTFEHLER: Fehler beim Herunterladen von S3 für den Bucket

Fehlerbedingung: STARTFEHLER: Fehler beim Herunterladen von S3 für den Bucket: `aws-glue-ml-data-quality-assets-us-east-1, key: jars/aws-glue-ml-data-quality-etl.jar.Access Denied (Service: Amazon S3; Status Code: 403; Please refer logs for details)` .

Lösung: Die Berechtigungen in der Rolle, die an AWS Glue Data Quality übergeben wurden, müssen das Lesen vom vorherigen Amazon S3 S3-Standort aus ermöglichen. Diese IAM-Richtlinie sollte der Rolle angefügt werden:

```
{
  "Sid": "allowS3",
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::aws-glue-ml-data-quality-assets-<region>/*"
}
```

Ausführliche Informationen zu den Berechtigungen finden Sie unter Data-Quality-Autorisierung. Diese Bibliotheken sind erforderlich, um die Datenqualität Ihrer Datensätze zu bewerten.

InvalidInputException (Status: 400): DataQuality Regeln können nicht analysiert werden

Fehlerzustand: InvalidInputException (Status: 400): DataQuality Regeln können nicht analysiert werden.

Lösung: Für diesen Fehler gibt es viele Ursachen. Eine Ursache besteht darin, dass Ihre Regeln einfache Anführungszeichen enthalten. Stellen Sie sicher, dass sie in doppelte Anführungszeichen gesetzt werden. Zum Beispiel:

```
Rules = [
ColumnValues "tipo_vinculo" in ["CODO", "DOCO", "COCO", "DODO"] AND "categoria" = 'ES'
    AND "cod_bandera" = 'CEP'
```

Ändern Sie dies in:

```
Rules = [
(ColumnValues "tipovinculo" in [ "CODO", "DOCO", "COCO", "DODO"]) AND (ColumnValues
 "categoria" = "ES")
    AND (ColumnValues "codbandera" = "CEP")
]
```

Fehler: Eventbridge löst basierend auf dem von mir eingerichteten Zeitplan keine Glue-DQ-Aufträge aus

Fehlerzustand: Eventbridge wird nicht ausgelöst AWS Glue Data Quality Jobs, die auf dem Zeitplan basieren, den ich eingerichtet habe.

Lösung: Möglicherweise verfügt die Rolle, die den Auftrag auslöst, nicht über die richtigen Berechtigungen. Stellen Sie sicher, dass die Rolle, die Sie zum Starten der Aufträge verwenden, über die in der IAM-Einrichtung genannten Berechtigungen verfügt, die zum Planen von Auswertungsausführungen erforderlich sind.

CustomSQL-Fehler

Fehlerbedingung: `The output from CustomSQL must contain at least one column that matches the input dataset for AWS Glue Data Quality to provide row level results. The SQL query is a valid query but no columns from the SQL result are present in the Input Dataset. Ensure that matching columns are returned from the SQL.`

Lösung: Die SQL-Abfrage ist gültig, Sie sollten aber darauf achten, nur Spalten aus der Primärtabelle auszuwählen. Die Auswahl von Aggregatfunktionen wie Summe und Anzahl für die Spalten der Primärdatenbank kann zu diesem Fehler führen.

Fehlerbedingung: `There was a problem when executing your SQL statement: cannot resolve "Col"`.

Lösung: Diese Spalte ist in der Primärtabelle nicht vorhanden.

Fehlerbedingung: `The columns that are returned from the SQL statement should only belong to the primary table. "In this case, some columns ( Col ) belong to reference table"`.

Lösung: Achten Sie bei SQL-Abfragen beim Verknüpfen der Primärtabelle mit anderen Referenztabellen darauf, dass Ihre Select-Anweisung nur Spaltennamen aus Ihrer Primärtabelle enthält, um Ergebnisse auf Zeilenebene für die Primärtabelle zu generieren.

Dynamische Regeln

Fehlerbedingung: `Dynamic rules require job context, and cannot be evaluated in interactive session or data preview..`

Ursache: Diese Fehlermeldung wird möglicherweise in den Ergebnissen Ihrer Datenvorschau oder in anderen interaktiven Sitzungen angezeigt, wenn Ihr Regelsatz dynamische DQ-Regeln enthält. Dynamische Regeln verweisen auf historische Metriken, die mit einem bestimmten Auftragsnamen

und Bewertungskontext verknüpft sind, sodass sie in interaktiven Sitzungen nicht ausgewertet werden können.

Lösung: Ausführen Ihres AWS Glue Der Job erzeugt historische Messwerte, auf die bei späteren Jobausführungen für denselben Job verwiesen werden kann.

Fehlerbedingung:

- `[RuleType] rule only supports simple atomic operands in thresholds.`.
- `Function last not yet implemented for [RuleType] rule.`

Lösung: Dynamische Regeln werden im Allgemeinen für alle DQDL-Regeltypen in numerischen Ausdrücken unterstützt (siehe DQDL-Referenz). Einige Regeln, die mehrere Metriken erzeugen ColumnLength, ColumnValues werden jedoch noch nicht unterstützt.

Fehlerbedingung: `Binary expression operands must resolve to a single number.`.

Ursache: Dynamische Regeln unterstützen binäre Ausdrücke wie `RowCount > avg(last(5)) * 0.9`. Hier ist der binäre Ausdruck `avg(last(5)) * 0.9`. Diese Regel ist gültig, weil beide Operanden (`avg(last(5))` und `0.9`) in eine einzige Zahl aufgelöst werden. Ein falsches Beispiel ist `RowCount > last(5) * 0.9`, weil `last(5)` eine Liste erzeugt, die nicht aussagekräftig mit der aktuellen Zeilenanzahl verglichen werden kann.

Lösung: Verwenden Sie Aggregationsfunktionen, um einen Operanden mit Listenwert auf eine einzige Zahl zu reduzieren.

Fehlerbedingung:

- `Rule threshold results in list, and a single value is expected. Use aggregation functions to produce a single value. Valid example: sum(last(10)), avg(last(10)).`
- `Rule threshold results in empty list, and a single value is expected.`

Ursache: Dynamische Regeln können verwendet werden, um einige Merkmale Ihres Datensatzes mit seinen historischen Werten zu vergleichen. Die letzte Funktion ermöglicht das Abrufen mehrerer historischer Werte, wenn ein positives Ganzzahl-Argument angegeben wird. `last(5)` ruft beispielsweise die fünf neuesten Werte ab, die bei Auftragsausführungen für Ihre Regel beobachtet wurden.

Lösung: Eine Aggregationsfunktion muss verwendet werden, um diese Werte auf eine einzige Zahl zu reduzieren und einen aussagekräftigen Vergleich mit dem in der aktuellen Auftragsausführung beobachteten Wert anzustellen.

Gültige Beispiele:

- `RowCount >= avg(last(5))`
- `RowCount > last(1)`
- `RowCount < last()`

Ungültiges Beispiel: `RowCount > last(5)`.

Fehlerbedingung:

- `Function index used in threshold requires positive integer argument.`
- `Index argument must be an integer. Valid syntax example: RowCount > index(last(10, 2)), which means RowCount must be greater than third most recent execution from last 10 job runs.`

Lösung: Beim Erstellen dynamischer Regeln können Sie die Aggregationsfunktion `index` verwenden, um einen historischen Wert aus einer Liste auszuwählen. Mit `RowCount > index(last(5)`, 1) wird beispielsweise geprüft, ob die im aktuellen Auftrag beobachtete Zeilenanzahl größer als die vorletzte Zeilenanzahl ist, die für den Auftrag beobachtet wurde. `index` hat einen Null-Index.

Fehlerbedingung: `IllegalArgumentException: Parsing Error: Rule Type: DetectAnomalies is not valid`.

Lösung: Die Erkennung von Anomalien ist nur verfügbar in AWS Glue 4.0.

Fehlerbedingung: `IllegalArgumentException: Parsing Error: Unexpected condition for rule of type ... no viable alternative at input ...`.

Hinweis: `...` ist dynamisch. Beispiel: `IllegalArgumentException: Parsing Error: Unexpected condition for rule of type RowCount with number return type, line 4:19 no viable alternative at input '>last'`.

Lösung: Die Erkennung von Anomalien ist nur verfügbar in AWS Glue 4.0.

Ausnahme in der Benutzerklasse: org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException

Zustand des Fehlers : `Exception in User Class: org.apache.spark.sql.AnalysisException: org.apache.hadoop.hive.ql.metadata.HiveException: Unable to fetch table mailpiece_submitted. StorageDescriptor#InputFormat cannot be null for table: mailpiece_submitted (Service: null; Status Code: 0; Error Code: null; Request ID: null; Proxy: null)`

Ursache: Sie verwenden Apache Iceberg in AWS Glue Data Catalog und das Eingabeformat-Attribut in AWS Glue Data Catalog ist leer.

Lösung: Dieses Problem tritt auf, wenn Sie den CustomSQL-Regeltyp in Ihrer DQ-Regel verwenden. Eine Möglichkeit, dieses Problem zu beheben, besteht darin, „primary“ zu verwenden oder einen Katalognamen hinzuzufügen. `glue_catalog.<database>.<table> in Custom ruletype`

UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing-Fehler: Es wurden keine Regeln oder Analysatoren bereitgestellt., keine praktikable Alternative bei der Eingabe

Zustand des Fehlers : `UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing Error: No rules or analyzers provided., no viable alternative at input`

Lösung: DQDL ist nicht analysierbar. Es gibt einige Fälle, in denen dies vorkommen kann. Wenn Sie zusammengesetzte Regeln verwenden, stellen Sie sicher, dass sie die richtigen Klammern haben.

```
(RowCount >= avg(last(10)) * 0.6) and (RowCount <= avg(last(10)) * 1.4) instead of
      RowCount >= avg(last(10)) * 0.6 and RowCount <= avg(last(10)) * 1.4
```

Amazon Q-Datenintegration in AWS Glue

Amazon Q Data Integration in AWS Glue ist eine neue generative KI-Funktion AWS Glue , die es Dateningenieuren und ETL-Entwicklern ermöglicht, Datenintegrationsjobs in natürlicher Sprache zu erstellen. Techniker und Entwickler können Amazon Q bitten, Jobs zu verfassen, Probleme zu beheben AWS Glue und Fragen zur Datenintegration zu beantworten.

Was ist Amazon Q?

> **Note**
>
> Bereitgestellt von Amazon Bedrock: AWS implementiert automatisierte Missbrauchserkennung. Da die Amazon Q-Datenintegration auf Amazon Bedrock basiert, können Benutzer die in Amazon Bedrock implementierten Kontrollen in vollem Umfang nutzen, um Sicherheit und den verantwortungsvollen Umgang mit künstlicher Intelligenz (KI) durchzusetzen.

Amazon Q ist ein auf generativer künstlicher Intelligenz (KI) basierender Konversationsassistent, der Ihnen helfen kann, AWS Anwendungen zu verstehen, zu erstellen, zu erweitern und zu betreiben. Das Modell, das Amazon Q zugrunde liegt, wurde um qualitativ hochwertige AWS Inhalte erweitert, damit Sie umfassendere, umsetzbare und referenzierte Antworten erhalten, sodass Sie schneller darauf AWS aufbauen können. Weitere Informationen finden Sie unter Was ist Amazon Q?

Was ist die Amazon-Q-Datenintegration in AWS Glue?

Die Amazon Q-Datenintegration AWS Glue umfasst die folgenden Funktionen:

- Chat — Amazon Q Data Integration in AWS Glue kann Fragen in natürlicher Sprache auf Englisch zu AWS Glue Datenintegrationsdomänen wie AWS Glue Quell- und Zielkonnektoren, AWS Glue ETL-Jobs, Datenkatalog, Crawlern und AWS Lake Formation anderen Funktionsdokumentationen und Best Practices beantworten. Amazon Q Data Integration in AWS Glue antwortet mit step-by-step Anweisungen und enthält Verweise auf seine Informationsquellen.
- Generierung von Datenintegrationscode — Amazon Q Data Integration in AWS Glue kann Fragen zu AWS Glue ETL-Skripts beantworten und neuen Code generieren, wenn eine Frage in natürlicher Sprache auf Englisch gestellt wird.

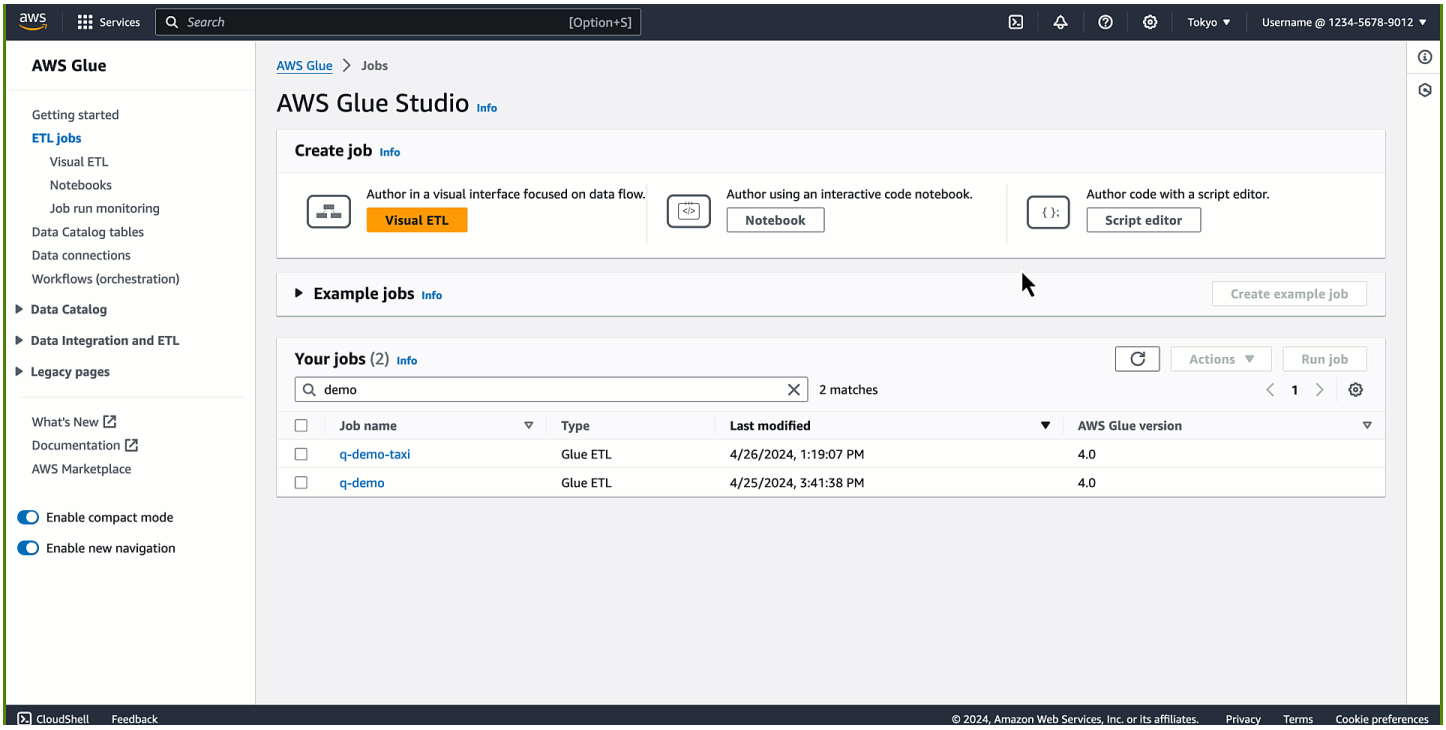
- Problembehandlung — Amazon Q Data Integration in AWS Glue wurde speziell entwickelt, um Ihnen zu helfen, Fehler bei AWS Glue Aufträgen zu verstehen, und bietet step-by-step Anweisungen zur Ursachenfindung und Lösung Ihrer Probleme.

> **Note**
>
> Amazon Q Data Integration in verwendet den Kontext Ihrer Konversation AWS Glue nicht als Grundlage für future Antworten für die Dauer Ihrer Konversation. Jede Konversation mit Amazon Q Data Integration in AWS Glue ist unabhängig von Ihren vorherigen oder future Konversationen.

Arbeiten Sie mit der Amazon-Q-Datenintegration in AWS Glue?

Im Amazon Q-Bereich können Sie Amazon Q auffordern, Code für ein AWS Glue ETL-Skript zu generieren, oder eine Frage zu AWS Glue Funktionen oder zur Behebung eines Fehlers beantworten. Die Antwort ist ein ETL-Skript PySpark mit step-by-step Anweisungen zur Anpassung des Skripts, zur Überprüfung und Ausführung. Bei Fragen wird die Antwort auf der Grundlage der Wissensdatenbank zur Datenintegration zusammen mit einer Zusammenfassung und einer Quell-URL als Referenz generiert.

Sie können Amazon Q beispielsweise bitten, "Bitte stellen Sie ein Glue-Skript bereit, das aus Snowflake liest, die Felder umbenennt und in Redshift schreibt"“. Als Antwort gibt Amazon Q Data Integration in AWS Glue ein AWS Glue Job-Skript zurück, das die angeforderte Aktion ausführen kann. Sie können den generierten Code überprüfen, um sicherzustellen, dass er die gewünschte Aufgabe erfüllt. Wenn Sie zufrieden sind, können Sie es als Auftrag in der Produktion einsetzen. AWS Glue Sie können Fehler in Aufträgen beheben, indem Sie die Integration fragen, Fehler und Ausfälle zu erklären und Lösungen vorzuschlagen. Amazon Q kann Fragen AWS Glue zu unseren Best Practices für die Datenintegration beantworten.



Im Folgenden finden Sie Beispielfragen, die zeigen, wie Amazon Q Data Integration in Ihnen beim Aufbau helfen AWS Glue kann AWS Glue:

AWS Glue Generierung von ETL-Code:

- Schreiben Sie ein AWS Glue Skript, das JSON aus S3 liest, Felder mithilfe von Apply Mapping transformiert und in Amazon Redshift schreibt
- Wie schreibe ich ein AWS Glue Skript zum Lesen aus DynamoDB, wende die DropNullFields Transformation an und schreibe als Parquet auf S3?
- Gib mir ein AWS Glue Skript, das aus MySQL liest, einige Felder basierend auf meiner Geschäftslogik löscht und in Snowflake schreibt
- Schreiben Sie einen AWS Glue Job zum Lesen aus DynamoDB und zum Schreiben in S3 als JSON
- Helfen Sie mir, ein AWS Glue Skript für AWS Glue Data Catalog to S3 zu entwickeln
- Schreiben Sie einen AWS Glue Job, um JSON aus S3 zu lesen, Nullen zu löschen und in Redshift zu schreiben

AWS Glue Erläuterungen zu den Funktionen:

- Wie verwende ich AWS Glue Data Quality?

- Wie benutzt man AWS Glue Job-Lesezeichen?
- Wie aktiviere ich AWS Glue Autoscaling?
- Was ist der Unterschied zwischen AWS Glue dynamischen Frames und Spark-Datenrahmen?
- Von welchen Verbindungsarten werden die verschiedenen Arten von Verbindungen unterstützt AWS Glue?

AWS Glue Problembehebung:

- Wie behebt man Out-of-Memory-Fehler (OOM) bei AWS Glue Jobs?
- Welche Fehlermeldungen werden möglicherweise beim Einrichten von AWS Glue Data Quality angezeigt, und wie können Sie diese beheben?
- Wie behebe ich einen AWS Glue Job mit dem Fehler Amazon S3 access denied?
- Wie löse ich Probleme mit dem Datenaustausch bei AWS Glue Aufträgen?

Bewährte Methoden für die Interaktion mit der Amazon Q-Datenintegration

Im Folgenden finden Sie bewährte Methoden für die Interaktion mit der Amazon Q-Datenintegration:

- Stellen Sie bei der Interaktion mit Amazon Q Data Integration spezifische Fragen, wiederholen Sie, wenn Sie komplexe Anfragen haben, und überprüfen Sie, ob die Antworten korrekt sind.
- Seien Sie bei der Bereitstellung von Eingabeaufforderungen zur Datenintegration in natürlicher Sprache so spezifisch wie möglich, damit der Assistent genau versteht, was Sie benötigen. Anstatt „Daten aus S3 extrahieren“ zu fragen, geben Sie mehr Details an, z. B. „Schreiben Sie ein AWS Glue Skript, das JSON-Dateien aus S3 extrahiert“.
- Überprüfen Sie das generierte Skript, bevor Sie es ausführen, um die Richtigkeit sicherzustellen. Wenn das generierte Skript Fehler enthält oder nicht Ihrer Absicht entspricht, geben Sie dem Assistenten Anweisungen zur Korrektur.
- Generative KI-Technologie ist neu und die Antworten können Fehler enthalten, die manchmal als Halluzinationen bezeichnet werden. Testen und überprüfen Sie den gesamten Code auf Fehler und Schwachstellen, bevor Sie ihn in Ihrer Umgebung oder Ihrem Workload verwenden.

Amazon Q-Datenintegration zur AWS Glue Serviceverbesserung

Um die Amazon Q-Datenintegration bei der AWS Glue Bereitstellung der relevantesten Informationen über AWS Services zu unterstützen, können wir bestimmte Inhalte von Amazon Q, wie z. B. Fragen, die Sie Amazon Q stellen, und deren Antworten, zur Serviceverbesserung verwenden.

Informationen darüber, welche Inhalte wir verwenden und wie Sie sich abmelden können, finden Sie unter Amazon Q Developer Service Improvement im Amazon Q Developer User Guide.

Überlegungen

Berücksichtigen Sie die folgenden Punkte, bevor Sie die Amazon-Q-Datenintegration in AWS Glue verwenden:

- Derzeit funktioniert die Codegenerierung nur mit dem PySpark Kernel. Der generierte Code ist für AWS Glue Jobs, die auf Python Spark basieren.
- Informationen zu den unterstützten Kombinationen von Codegenerierungsfunktionen der Amazon Q-Datenintegration in AWS Glue finden Sie unterUnterstützte Funktionen zur Codegenerierung.

Einrichtung der Amazon Q-Datenintegration in AWS Glue

Die folgenden Abschnitte enthalten Informationen über die Einrichtung der Amazon-Q-Datenintegration in AWS Glue.

Themen

- Konfigurieren von IAM-Berechtigungen

Konfigurieren von IAM-Berechtigungen

In diesem Thema werden die IAM-Berechtigungen beschrieben, die Sie für das Amazon Q-Chat-Erlebnis und das AWS Glue Studio-Notebook-Erlebnis konfigurieren.

Themen

- Konfiguration von IAM-Berechtigungen für Amazon Q Chat
- Konfiguration von IAM-Berechtigungen für Studio-Notebooks AWS Glue

Konfiguration von IAM-Berechtigungen für Amazon Q Chat

Die Erteilung von Berechtigungen für die von Amazon Q APIs verwendete Datenintegration AWS Glue erfordert entsprechende AWS Identity and Access Management (IAM) -Berechtigungen. Sie können Berechtigungen erhalten, indem Sie Ihrer IAM-Identität (z. B. einem Benutzer, einer Rolle oder einer Gruppe) die folgende benutzerdefinierte AWS Richtlinie zuordnen:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartCompletion",
                "glue:GetCompletion"
            ],
            "Resource": [
                "arn:aws:glue:*:*:completion/*"
            ]
        }
    ]
}
```

Konfiguration von IAM-Berechtigungen für Studio-Notebooks AWS Glue

Um die Amazon Q-Datenintegration in AWS Glue Studio-Notebooks zu aktivieren, stellen Sie sicher, dass der Notebook-IAM-Rolle die folgende Berechtigung zugewiesen ist:

> **Note**
>
> Das `codewhisperer` Präfix ist ein älterer Name eines Dienstes, der mit Amazon Q Developer zusammengeführt wurde. Weitere Informationen finden Sie unter Amazon Q Developer rename — Zusammenfassung der Änderungen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartCompletion",
                "glue:GetCompletion"
            ],
            "Resource": [
                "arn:aws:glue:*:*:completion/*"
            ]
        },
        {
            "Sid": "AmazonQDeveloperPermissions",
            "Effect": "Allow",
            "Action": [
                "codewhisperer:GenerateRecommendations"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Die Amazon Q-Datenintegration ist AWS Glue nicht über das AWS SDK APIs verfügbar, das Sie programmgesteuert verwenden können. Die folgenden beiden APIs werden in der IAM-Richtlinie verwendet, um dieses Erlebnis über das Amazon Q-Chat-Panel oder AWS Glue Studio-Notizbücher zu aktivieren: `StartCompletion` und`GetCompletion`.

Zuweisen von Berechtigungen

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in AWS IAM Identity Center: Erstellen Sie einen Berechtigungssatz. Folgen Sie den Anweisungen unter Einen Berechtigungssatz erstellen im AWS IAM Identity Center-Benutzerhandbuch.
- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden: Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter Erstellen einer Rolle für einen externen Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch.
- IAM-Benutzer:
  - Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter Erstellen einer Rolle für einen IAM-Benutzer im IAM-Benutzerhandbuch.
  - (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter Hinzufügen von Berechtigungen zu einem Benutzer (Konsole) im IAM-Benutzerhandbuch.

Unterstützte Funktionen zur Codegenerierung

Im Folgenden sind die Kombinationen der Codegenerierungsfunktionen der Amazon Q-Datenintegration aufgeführt.

Quellen und Ziele	Transformation
S3 mit den folgenden Formattypen: json, csv, parquet, hudi, delta	Entfernen
AWS Glue Data Catalog	Aggregate
Redlake	DropDuplicates
Amazon DynamoDB	Join
MySQL	Filter
Oracle	RenameColumns
PostgresSQL	FillNull
Microsoft SQL Server	DropNull
Amazon DocumentDB//MongoDB	WithColumns

Quellen und Ziele	Transformation
Snowflake	SQL-Abfrage
Google BigQuery	Union
Teradata	Select
OpenSearch Amazon-Dienst	
Vertica	
SAP HANA	
Amazon Redshift	

Beispiele für Interaktionen

Mit der Amazon Q-Datenintegration AWS Glue können Sie Ihre Frage im Amazon Q-Panel eingeben. Sie können eine Frage zur Datenintegrationsfunktion eingeben, die von bereitgestellt wird AWS Glue. Sie erhalten eine ausführliche Antwort zusammen mit Referenzdokumenten.

Ein weiterer Anwendungsfall ist die Generierung von AWS Glue ETL-Jobskripten. Sie können eine Frage zur Ausführung eines Datenextraktions-, Transformations- und Ladejobs stellen. Ein generiertes PySpark Skript wird zurückgegeben.

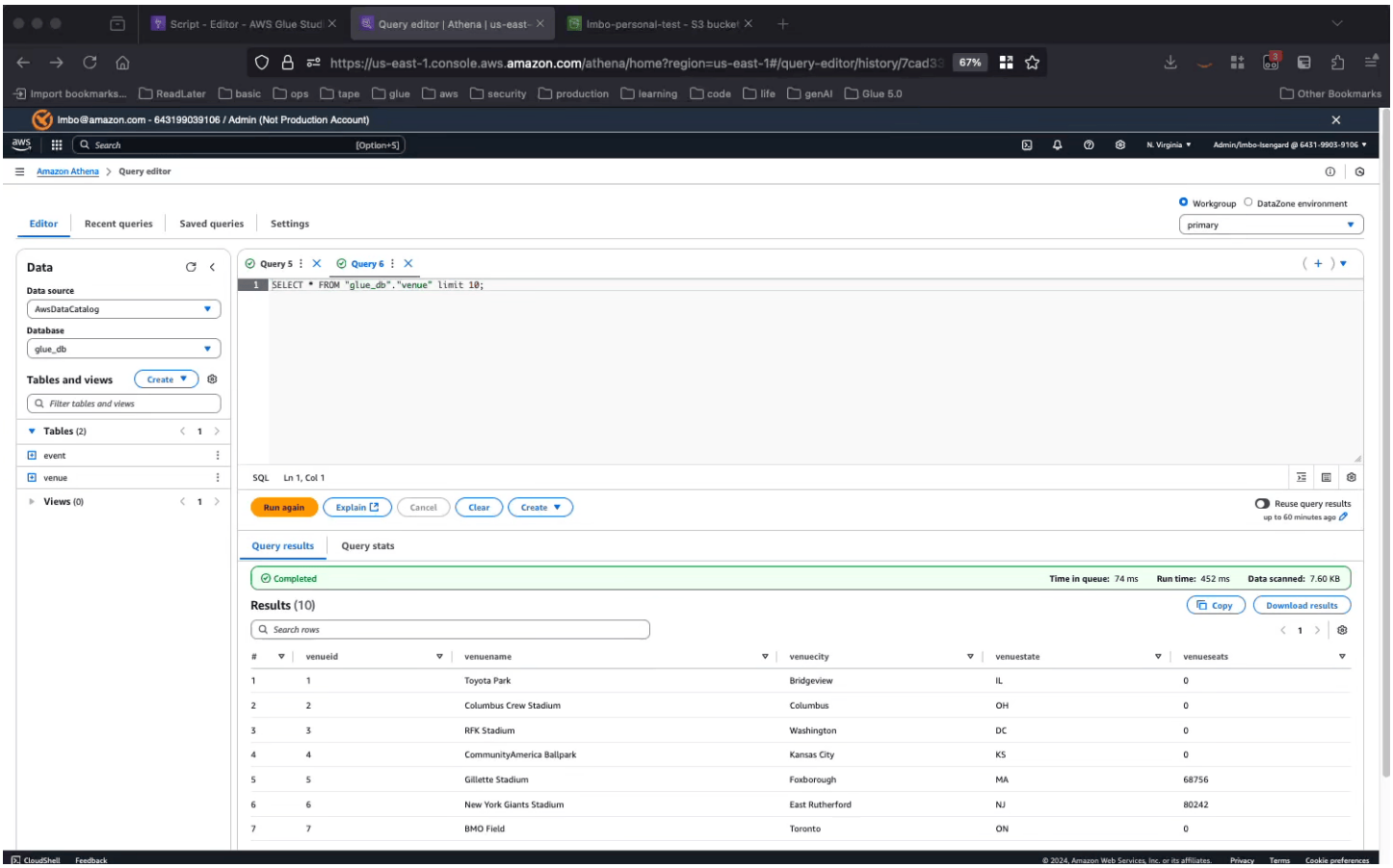
Themen

- Amazon Q-Chat-Interaktionen
- AWS Glue Interaktionen mit Studio-Notebooks

Amazon Q-Chat-Interaktionen

Beginnen Sie auf der AWS Glue Konsole mit der Erstellung eines neuen Jobs und fragen Sie Amazon F: „Erstellen Sie einen Glue-ETL-Flow, stellen Sie eine Verbindung zu zwei Glue-Katalogtabellen für Veranstaltungsort und Veranstaltung in meiner Datenbank glue_db her, verknüpfen Sie die Ergebnisse mit der Venueid des Veranstaltungsorts und der e_venueid der Veranstaltung, filtern Sie dann nach dem Veranstaltungsortstatus mit der Bedingung

VenueState=='dc' und schreiben Sie im CSV-Format an s3://amzn-s3-demo -/."“ bucket/codegen/BDB-9999/output



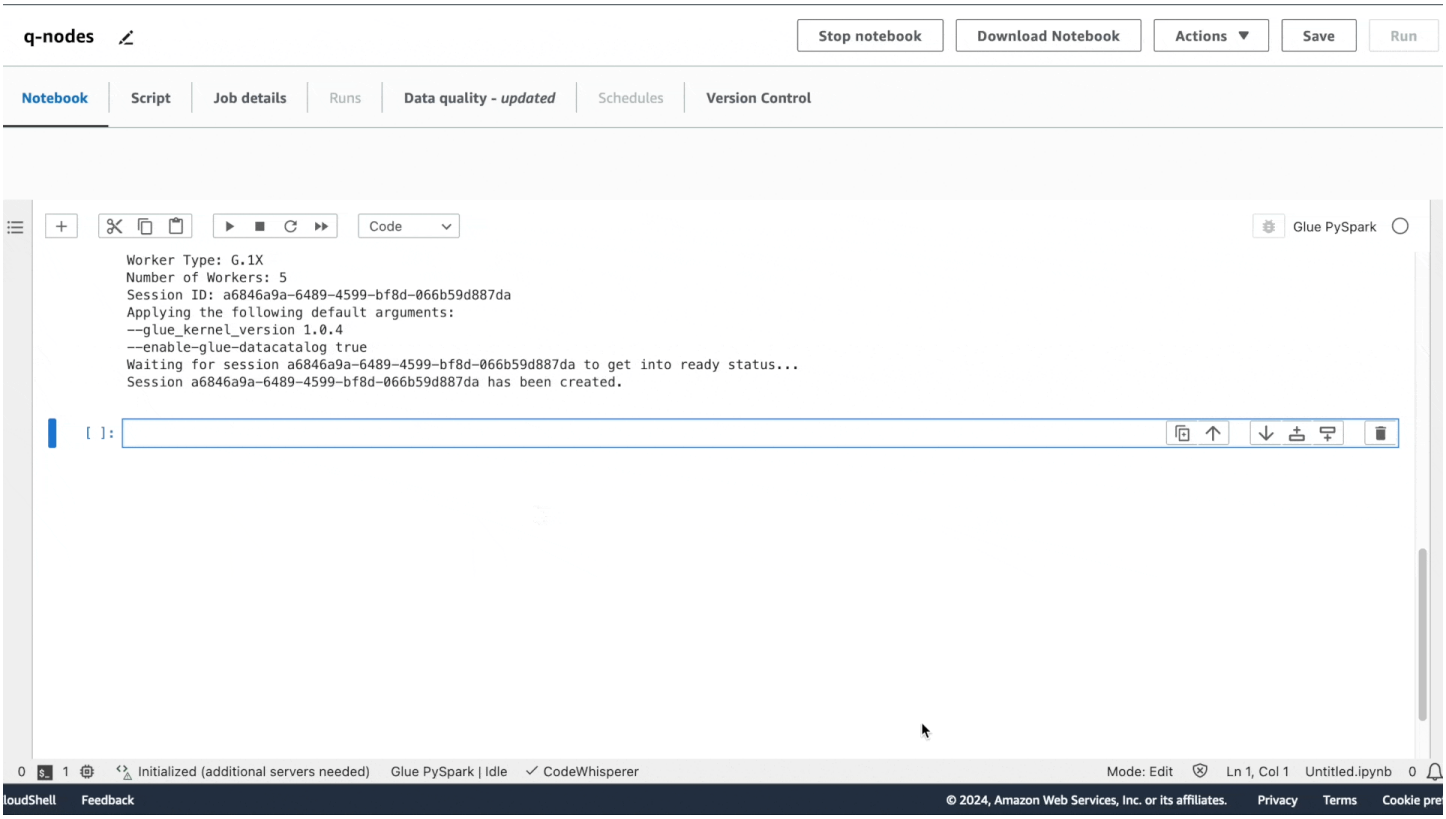
Sie werden feststellen, dass der Code generiert wurde. Mit dieser Antwort können Sie lernen und verstehen, wie Sie AWS Glue Code für Ihren Zweck verfassen können. Sie können den generierten Code kopieren/in den Skripteditor einfügen und Platzhalter konfigurieren. Nachdem Sie eine IAM-Rolle und AWS Glue Verbindungen für den Job konfiguriert haben, speichern Sie den Job und führen Sie ihn aus. Wenn der Job abgeschlossen ist, können Sie überprüfen, ob die Übersichtsdaten erwartungsgemäß in Amazon S3 gespeichert werden und von Ihren nachgelagerten Workloads verwendet werden können.

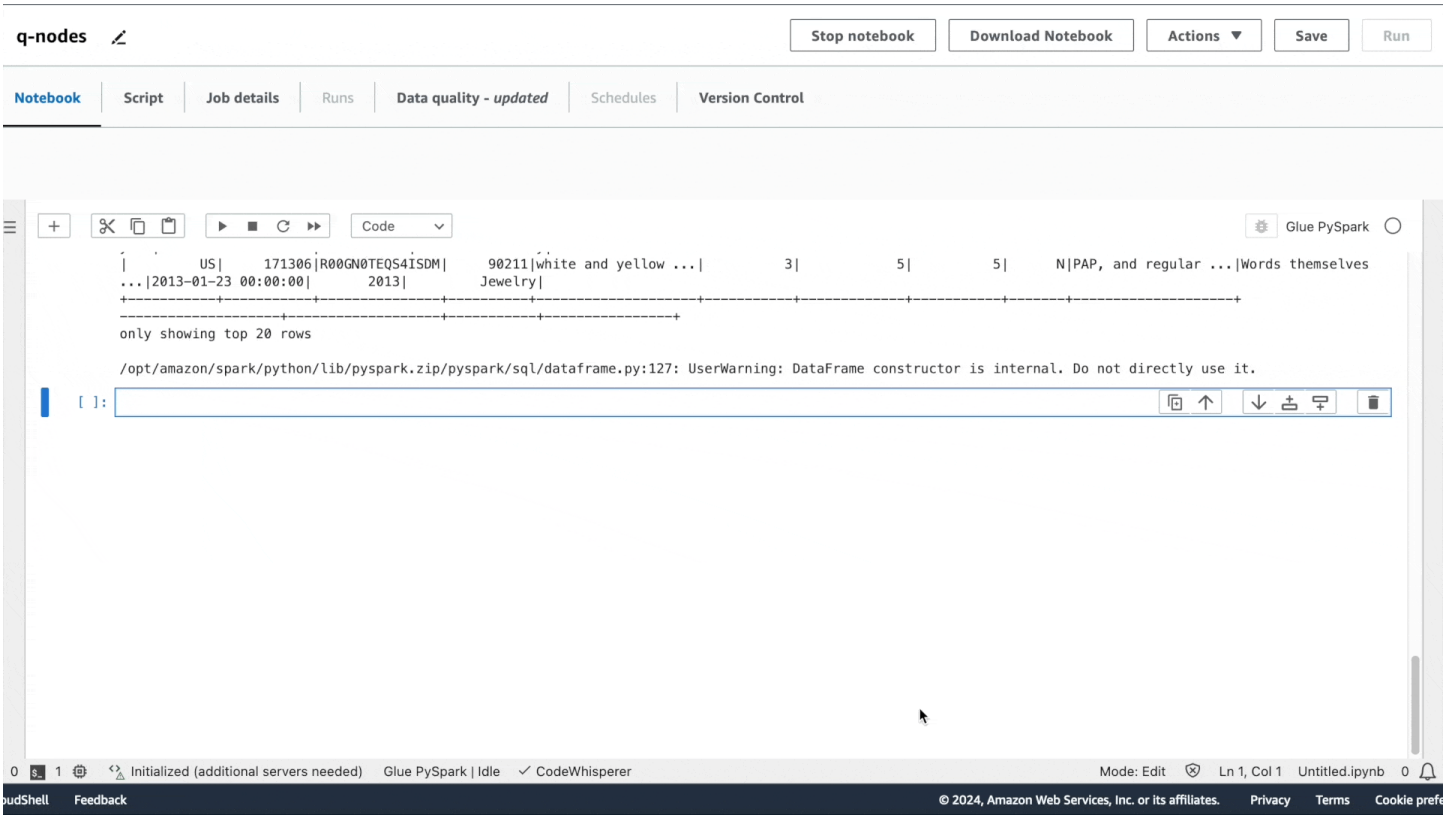
AWS Glue Interaktionen mit Studio-Notebooks

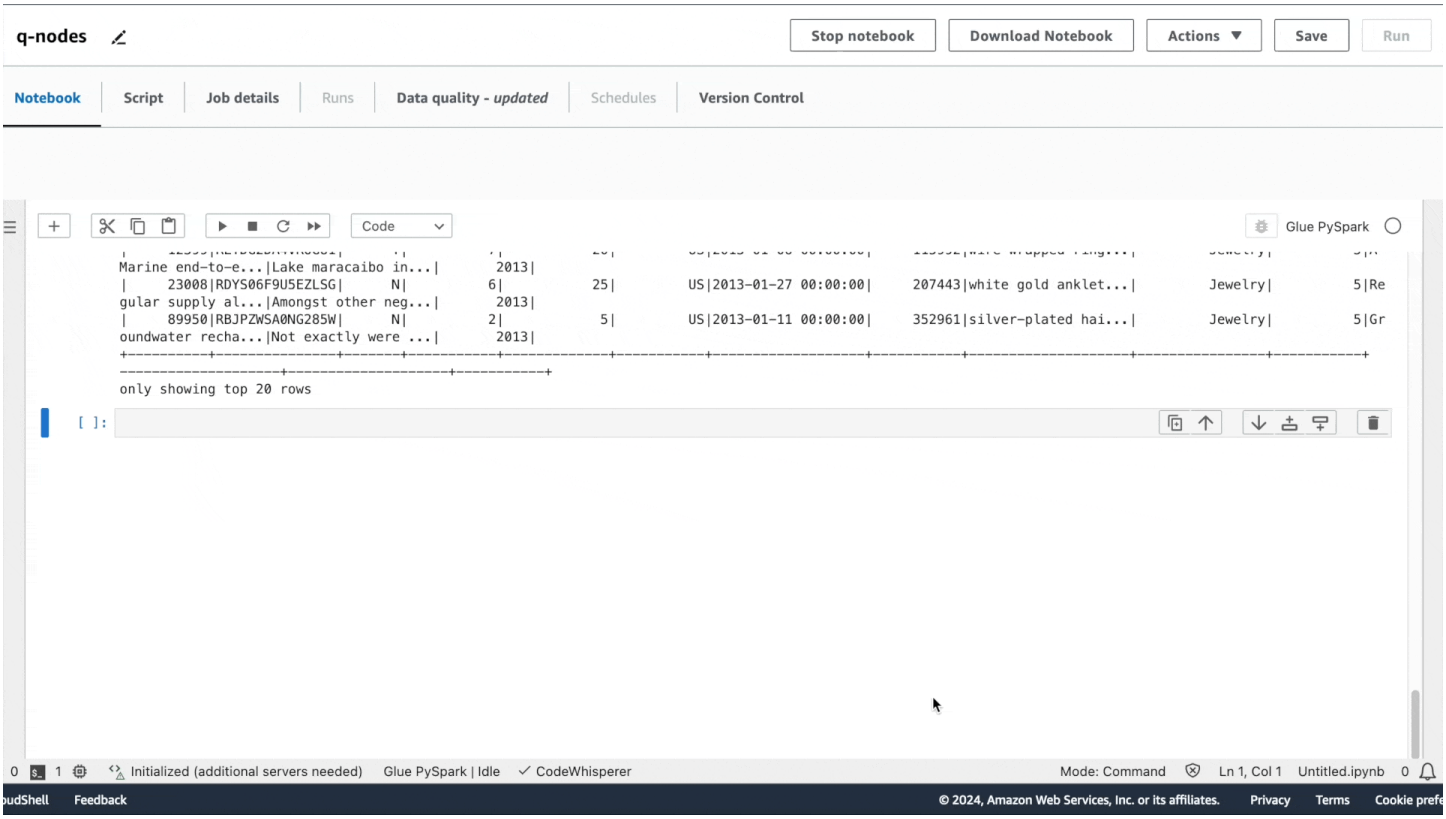
> **Note**
>
> Die Erfahrung mit Amazon Q Data Integration in AWS Glue Studio Notebooks konzentriert sich immer noch auf den DynamicFrame basierten Datenintegrationsfluss.

Fügen Sie eine neue Zelle hinzu und geben Sie Ihren Kommentar ein, um zu beschreiben, was Sie erreichen möchten. Nachdem Sie die Tabulatortaste und die Eingabetaste gedrückt haben, wird der empfohlene Code angezeigt.

Die erste Absicht besteht darin, die Daten zu extrahieren: „Gib mir Code, der eine Glue Data Catalog-Tabelle liest“, gefolgt von „Gib mir Code, um eine Filtertransformation mit star_rating>3" anzuwenden und „Gib mir Code, der den Frame als Parquet in S3 schreibt“.




q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
only showing top 20 rows
/opt/amazon/spark/python/lib/pyspark.zip/pyspark/sql/dataframe.py:127: UserWarning: DataFrame constructor is internal. Do not directly use it.
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Edit
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms


q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
only showing top 20 rows
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Command
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms

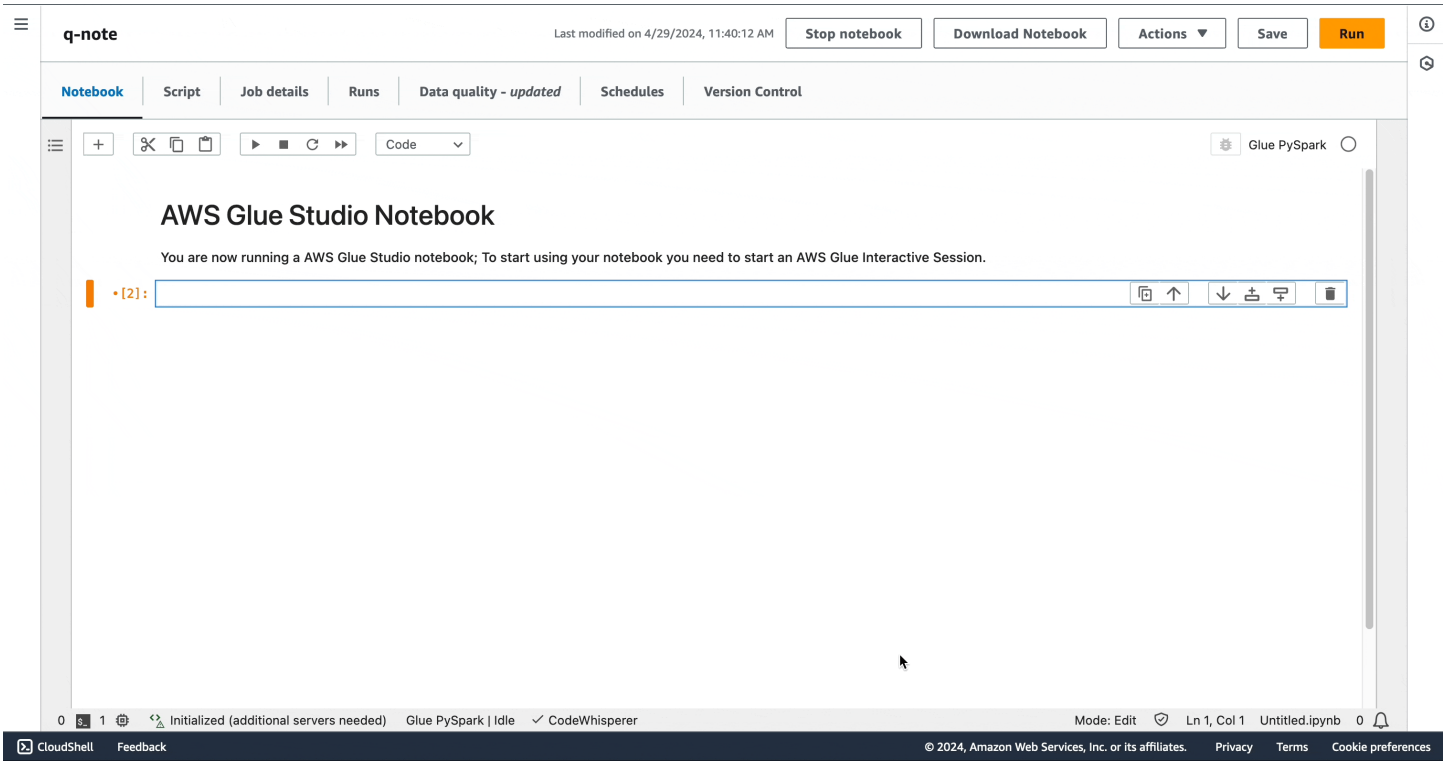
Ähnlich wie beim Amazon Q-Chat-Erlebnis wird der Code empfohlen. Wenn Sie die Tabulatortaste drücken, wird der empfohlene Code ausgewählt.

Sie können jede Zelle ausführen, indem Sie die entsprechenden Optionen für Ihre Quellen in den generierten Code eingeben. Mit der `show()` Methode können Sie zu jedem Zeitpunkt der Rechenläufe auch eine Vorschau einer Stichprobe Ihres Datensatzes anzeigen.

Sie können das Notizbuch als Job ausführen, entweder programmgesteuert oder indem Sie Ausführen wählen.

Komplexe Eingabeaufforderungen

Sie können ein vollständiges Skript mit einer einzigen komplexen Eingabeaufforderung generieren. „Ich habe JSON-Daten in S3 und Daten in Oracle, die kombiniert werden müssen. Bitte stellen Sie ein Glue-Skript bereit, das aus beiden Quellen liest, einen Join durchführt und dann die Ergebnisse in Redshift schreibt.“



Möglicherweise stellen Sie fest, dass die Amazon Q-Datenintegration auf dem Notizbuch denselben Codeausschnitt AWS Glue generiert hat, der im Amazon Q-Chat generiert wurde.

Sie können das Notizbuch als Job ausführen, indem Sie entweder Ausführen oder programmgesteuert wählen.

Verwendung von Kontexterkennung mit Amazon Q Data Integration in AWS Glue

Mit abfragebasierter, kontextsensitiver und PySpark DataFrame Codegenerierung in Amazon Q Data Integration können Sie Datenverarbeitungsaufträge jetzt effizienter erstellen. Sie können diese Aufforderung beispielsweise verwenden, um PySpark Code zu generieren: „Erstellen Sie einen Job zum Laden von Verkaufsdaten aus der Redshift-Tabelle 'analytics.salesorder' mithilfe der Verbindung 'erp_conn', filtern Sie order_amount unter 50 Dollar und speichern Sie sie in Amazon S3 im Parquet-Format.“

Amazon Q generiert das Skript auf der Grundlage Ihrer Eingabeaufforderung und der Einrichtung des Workflows zur Datenintegration mit den in Ihrer Frage angegebenen Details, wie Verbindungskonfigurationen, Schemadetails, Datenbank-/Tabellennamen und Spaltenspezifikationen für Transformationen. Vertrauliche Informationen, wie Passwörter für Verbindungsoptionen, werden weiterhin geschwärzt.

Wenn Sie die erforderlichen Informationen in Ihrer Aufforderung nicht angeben, fügt Amazon Q Platzhalter ein. Sie müssen den generierten Code mit den entsprechenden Werten aktualisieren, bevor Sie den Code ausführen.

Im Folgenden finden Sie Beispiele für die Verwendung von Context Awareness.

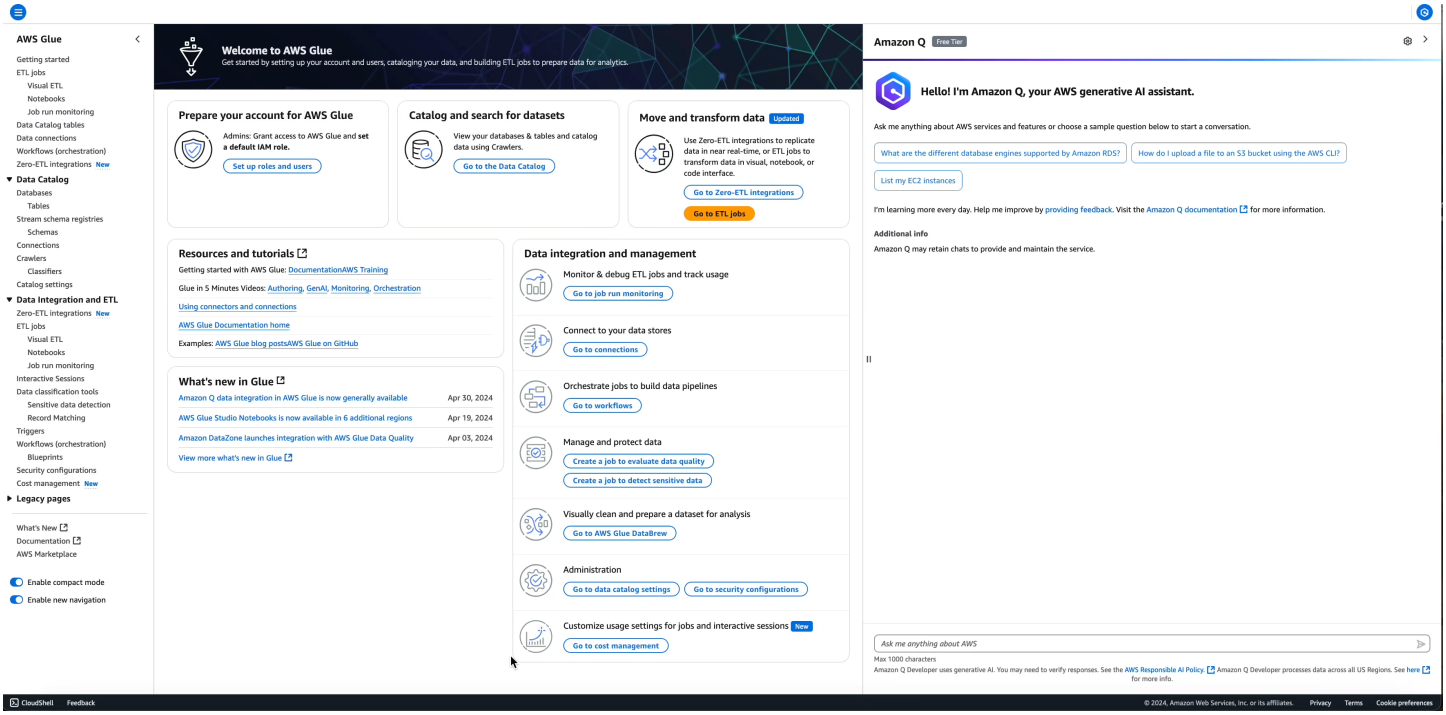
Beispiel: Interaktionen

Eine Amazon S3 S3-Datei lesen und in einen Amazon S3 S3-Bucket schreiben

Aufforderung: `Create an AWS Glue spark job that reads a file s3://amzn-s3-demo-bucket-input-bucket-name/input-filename.csv and saves it into the parquet file under directory s3://amzn-s3-demo-bucket-output-bucket-name/output-s3-prefix/`

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
from pyspark.sql import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node S3DataSource
```

```
S3DataSource_dsource1 = spark.read.format("csv").load(
    "s3://amzn-s3-demo-bucket-input-bucket-name/input-filename.csv"
)

# Script generated for node S3DataSink
S3DataSource_dsource1.write.format("parquet").mode("append").save(
    "s3://amzn-s3-demo-bucket-output-bucket-name/output-s3-prefix/"
)
```



Holen Sie sich Daten von Lakehouse und schreiben Sie sie in die Datenbank

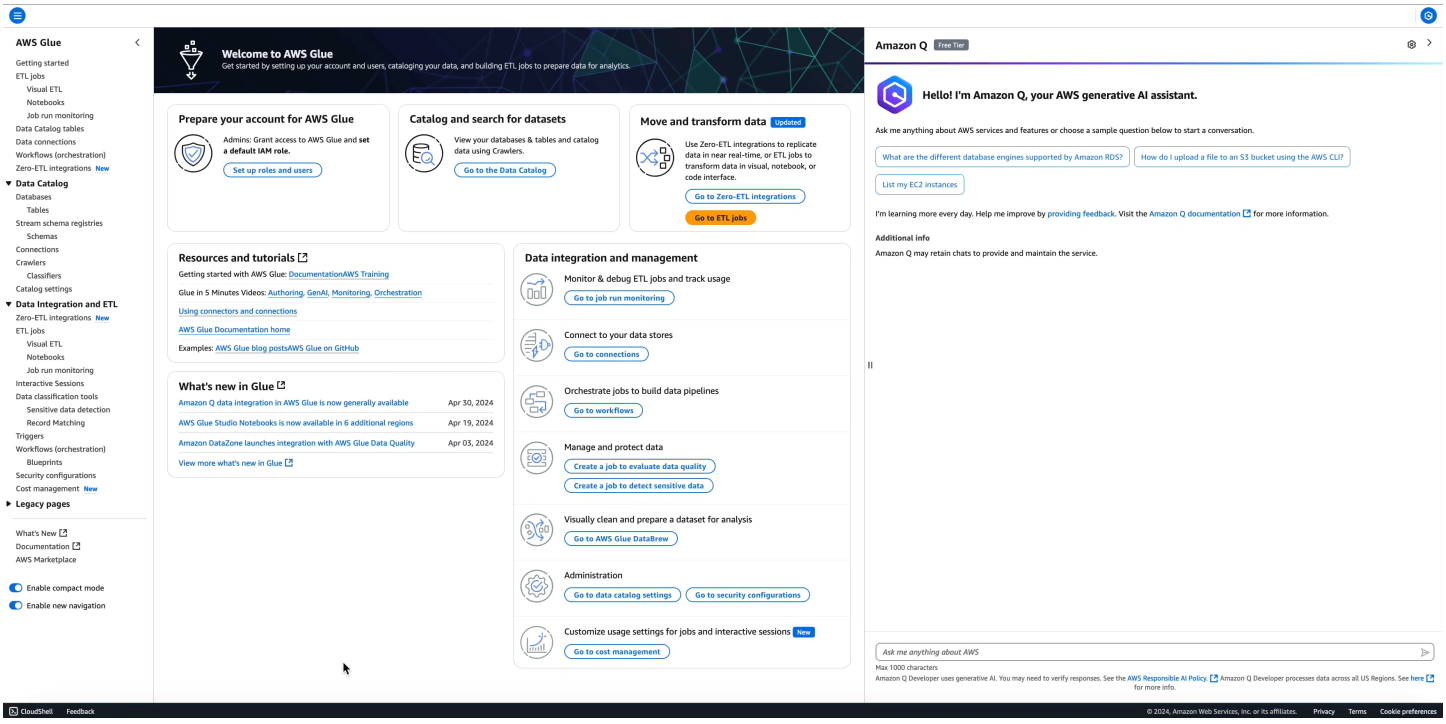
Aufforderung: `write an ETL script to read from a Lakehouse table my-table in database my-database and write it to a RDS MySQL table my-target-table`

Für Felder, in denen Sie keine Informationen angegeben haben (der erforderliche ConnectionName ist beispielsweise für die MySQL-Datensenke und standardmäßig mit einem Platzhalter <connection-name>im generierten Code), wird ein Platzhalter beibehalten, damit Sie die erforderlichen Informationen eingeben können, bevor Sie das Skript ausführen.

Generiertes Skript:

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
```

```
from pyspark.sql import SparkSession
from connectivity.adapter import CatalogConnectionHelper

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node S3DataSource
S3DataSource_dsource1 = spark.read.format("parquet").load(
    "s3://amzn-lakehouse-demo-bucket/my-database/my-table"
)

# Script generated for node ConnectionV2DataSink
ConnectionV2DataSink_dsink1_additional_options = {"dbtable": "my-target-table"}
CatalogConnectionHelper(spark).write(
    S3DataSource_dsource1,
    "mysql",
    "<connection-name>",
    ConnectionV2DataSink_dsink1_additional_options,
)
```



Beispiel: Vollständiger ETL-Workflow

Das folgende Beispiel zeigt, wie Sie AWS Glue bitten können, ein AWS Glue-Skript zu erstellen, um einen vollständigen ETL-Workflow mit der folgenden Aufforderung abzuschließen:`Create a`

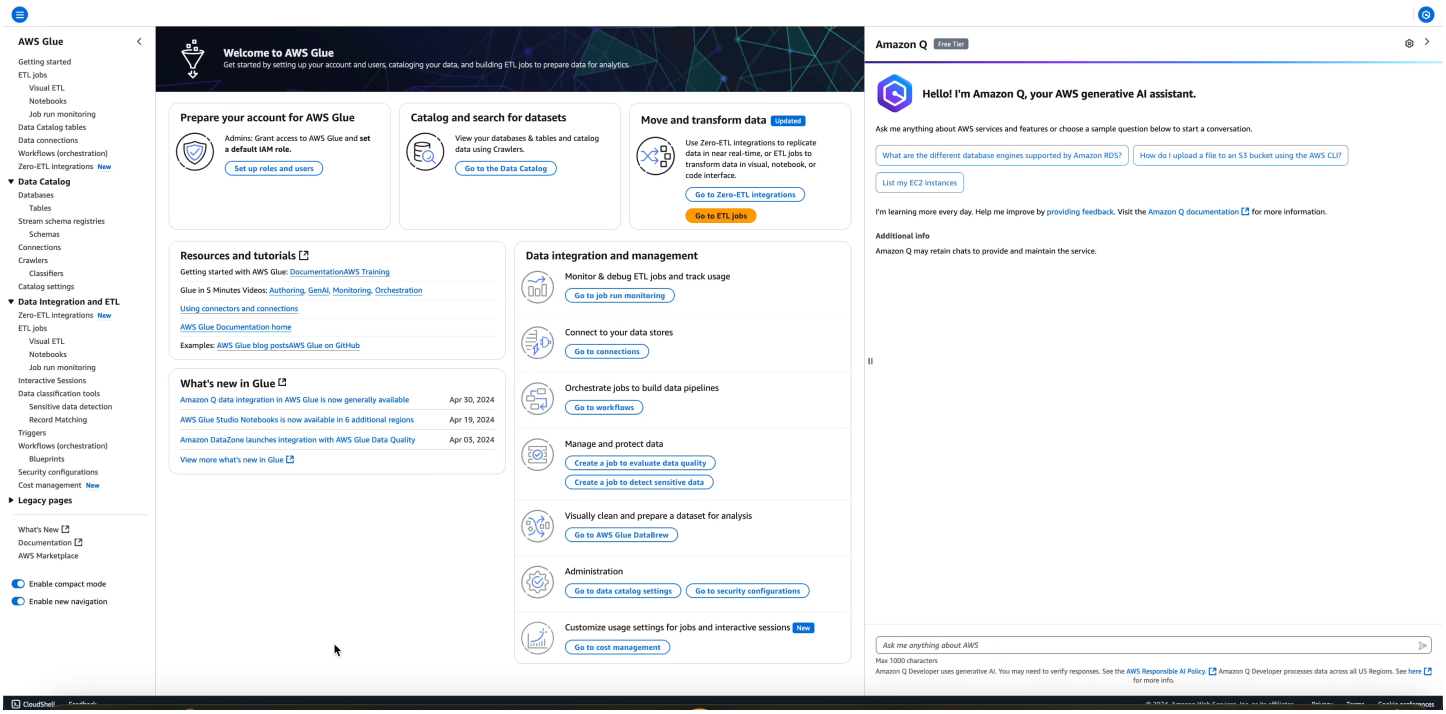
`AWS Glue ETL Script read from two AWS Glue Data Catalog tables venue and event in my database glue_db_4fthqih3vvk1if, join the results on the field venueid, filter on venue state with condition as venuestate=='DC' after joining the results and write output to an Amazon S3 S3 location s3://amz-s3-demo-bucket/output/ in CSV format.`

Der Workflow umfasst das Lesen aus verschiedenen Datenquellen (zwei AWS Glue Data Catalog-Tabellen) und einige Transformationen nach dem Lesen, indem das Ergebnis aus zwei Lesungen zusammengeführt, nach bestimmten Bedingungen gefiltert und die transformierte Ausgabe im CSV-Format an ein Amazon S3 S3-Ziel geschrieben wird.

Der generierte Job füllt die detaillierten Informationen zur Datenquelle, Transformation und Senken mit den entsprechenden Informationen aus der Benutzerfrage aus (siehe unten).

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
from pyspark.sql import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node CatalogDataSource
CatalogDataSource_dsource1 = spark.sql("select * from
 `glue_db_4fthqih3vvk1if`.`venue`")

# Script generated for node CatalogDataSource
CatalogDataSource_dsource2 = spark.sql("select * from
 `glue_db_4fthqih3vvk1if`.`event`")

# Script generated for node JoinTransform
JoinTransform_transform1 = CatalogDataSource_dsource1.join(
    CatalogDataSource_dsource2,
    (CatalogDataSource_dsource1["venueid"] == CatalogDataSource_dsource2["venueid"]),
    "inner",
)

# Script generated for node FilterTransform
FilterTransform_transform2 = JoinTransform_transform1.filter("venuestate=='DC'")

# Script generated for node S3DataSink
```

```
FilterTransform_transform2.write.format("csv").mode("append").save(
    "s3://amz-s3-demo-bucket/output//output/"
)
```



Einschränkungen

- Übertragung des Kontextes:
  - Die Funktion zur Kontexterkennung überträgt nur den Kontext aus der vorherigen Benutzerabfrage innerhalb derselben Konversation. Der Kontext, der über die unmittelbar vorhergehende Abfrage hinausgeht, wird nicht beibehalten.
- Support für Knotenkonfigurationen:
  - Derzeit unterstützt Context-Awareness nur eine Teilmenge der erforderlichen Konfigurationen für verschiedene Knoten.
  - Die Support optionaler Felder ist in kommenden Versionen geplant.
- Verfügbarkeit:
  - Kontextsensitivität und DataFrame Support sind in Q Chat- und SageMaker Unified Studio-Notebooks verfügbar. Diese Funktionen sind jedoch in AWS Glue Studio-Notebooks noch nicht verfügbar.

Orchestrierung in AWS Glue

In den folgenden Abschnitten erhalten Sie Informationen zum Orchestrierung von Aufträgen in AWS Glue.

Themen

- Starten von Aufträgen und Crawlern über Auslöser
- Durchführung komplexer ETL-Aktivitäten mithilfe von Blueprints und Workflows in AWS Glue
- Entwicklung von Bauplänen in AWS Glue

Starten von Aufträgen und Crawlern über Auslöser

In AWS Glue, können Sie Datenkatalogobjekte, sogenannte Trigger, erstellen, die Sie verwenden können, um entweder manuell oder automatisch einen oder mehrere Crawler zu starten oder Aufträge (ETL) zu extrahieren, zu transformieren und zu laden. Mithilfe von Auslösern können Sie eine Kette von abhängigen Aufträgen und Crawlern entwerfen.

> **Note**
>
> Sie können dasselbe erreichen, indem Sie Workflows definieren. Für die Erstellung komplexer ETL-Operationen mit mehreren Aufträgen werden Workflows bevorzugt. Weitere Informationen finden Sie unter the section called “Ausführen von komplexen ETL-Aktivitäten mithilfe von Blueprints und Workflows”.

Themen

- AWS Glue triggers
- Hinzufügen von Auslösern
- Aktivieren und Deaktivieren von Auslösern

AWS Glue triggers

Wenn ein Auslöser ausgelöst wird, kann er angegebene Aufträge und Crawler starten. Ein Auslöser wird nach Bedarf, basierend auf einem Zeitplan oder basierend auf einer Kombination von Ereignissen ausgelöst.

> **Note**
>
> Nur zwei Crawler können durch einen einzigen Auslöser aktiviert werden. Wenn Sie das Crawling für mehrere Datenspeicher durchführen möchten, verwenden Sie mehrere Quellen für jeden Crawler, anstatt mehrere Crawler gleichzeitig auszuführen.

Ein Auslöser kann einen von mehreren Zuständen besitzen. Ein Auslöser ist entweder `CREATED`, `ACTIVATED` oder `DEACTIVATED`. Es gibt auch Übergangszustände, wie z. B. `ACTIVATING`. Um einen Auslöser vorübergehend zu stoppen, können Sie ihn deaktivieren. Sie können ihn dann später wieder aktivieren.

Es gibt drei Arten von Auslösern:

Scheduled (Geplant)

Ein zeitbasierter Auslöser basierend auf `cron`.

Sie können einen Auslöser für eine Gruppe von Aufträgen oder Crawlern basierend auf einem Zeitplan erstellen. Sie können Einschränkungen angeben, z. B. mit welcher Häufigkeit Aufträge oder Crawler ausgeführt werden sowie an welchen Wochentagen und zu welcher Uhrzeit sie ausgeführt werden. Diese Einschränkungen basieren auf `cron`. Wenn Sie einen Zeitplan für einen Auslöser aufstellen, berücksichtigen Sie die Features und Einschränkungen von cron. Wenn Sie z. B. Ihren Crawler jeden Monat am 31. ausführen möchten, denken Sie daran, dass einige Monate keine 31 Tage haben. Weitere Informationen zu cron finden Sie unter Zeitpläne für Aufträge und Crawler.

Bedingt

Ein Auslöser, der ausgelöst wird, wenn ein vorheriger Auftrag oder Crawler oder mehrere Aufträge oder Crawler eine Liste von Bedingungen erfüllen.

Wenn Sie einen bedingten Auslöser erstellen, geben Sie eine Liste der Aufträge und eine Liste der Crawlern an, die zu beobachten sind. Für jeden überwachten Auftrag oder Crawler geben Sie einen zu überwachenden Status an, z. B. erfolgreich, fehlgeschlagen, Zeitüberschreitung usw. Der Auslöser wird ausgelöst, wenn die überwachten Aufträge oder Crawler im angegebenen Zustand enden. Sie können den Auslöser so konfigurieren, dass er ausgelöst wird, wenn eines oder alle der überwachten Ereignisse eintreten.

Beispielsweise können Sie den Auslöser T1 zum Starten von Auftrag J3 konfigurieren, wenn sowohl Auftrag J1 als auch Auftrag J2 erfolgreich abgeschlossen wurden, und den weiteren Auslöser T2, um Auftrag J4 zu starten, wenn entweder Auftrag J1 oder Auftrag J2 fehlschlägt.

In der folgenden Tabelle sind die Auftrags- und Crawler-Abschlusszustände (Ereignisse) aufgeführt, nach denen Auslöser suchen.

Auftragsabschlusszustände	Crawler-Abschlusszustände
• `SUCCEEDED` • `STOPPED` • `FAILED` • `TIMEOUT`	• `SUCCEEDED` • `FAILED` • `CANCELLED`

On-Demand-Modus

Ein Auslöser, der ausgelöst wird, wenn Sie ihn aktivieren. On-Demand-Auslöser gelangen niemals in den Zustand `ACTIVATED` oder `DEACTIVATED`. Sie verbleiben immer im Zustand `CREATED`.

Damit sie sofort ausgelöst werden können, sobald sie vorhanden sind, können Sie ein Flag zum Aktivieren geplanter und bedingter Auslöser beim Erstellen durch Sie festlegen.

> **Important**
>
> Aufträge oder Crawler, die bedingt durch andere Aufträge oder Crawler ausgeführt werden, werden als abhängig bezeichnet. Abhängige Aufträge oder Crawler werden nur gestartet, wenn der abschließende Auftrag oder Crawler durch einen Auslöser gestartet wurde. Alle Aufträge oder Crawler in einer Abhängigkeitskette müssen abhängige Elemente eines einzelnen Schedule (Zeitplan)- oder on-demand (On-Demand)-Auslösers sein.

Übergeben von Auftragsparametern mit Auslösern

Ein Auslöser kann Parameter an die Aufträge übergeben, die durch ihn gestartet werden. Zu den Parametern gehören Auftragsargumente, Timeout-Wert, Sicherheitskonfiguration und mehr. Wenn der Auslöser mehrere Aufträge startet, werden die Parameter an jeden Auftrag übergeben.

Im Folgenden werden die Regeln für Auftragsargumente aufgeführt, die durch einen Auslöser übergeben werden:

- Wenn der Schlüssel im Schlüssel-Wert-Paar mit einem Standard-Auftragsargument übereinstimmt, überschreibt das übergebene Argument das Standardargument. Wenn der Schlüssel mit keinem Standardargument übereinstimmt, wird das Argument als zusätzliches Argument an den Auftrag übergeben.
- Wenn der Schlüssel im Schlüssel-Wert-Paar mit einem nicht überschreibbaren Argument übereinstimmt, wird das übergebene Argument ignoriert.

Weitere Informationen finden Sie the section called “Auslöser” in der AWS Glue API.

Hinzufügen von Auslösern

Sie können einen Trigger hinzufügen, indem Sie AWS Glue console, die AWS Command Line Interface (AWS CLI) oder die AWS Glue API.

> **Note**
>
> Derzeit ist die AWS Glue Die Konsole unterstützt bei der Arbeit mit Triggern nur Jobs, keine Crawler. Sie können das AWS CLI oder verwenden AWS Glue API zur Konfiguration von Triggern sowohl für Jobs als auch für Crawler.

So fügen Sie einen Auslöser hinzu (Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter ETL die Option Triggers (Auslöser) aus. Wählen Sie dann Add trigger (Auslöser hinzufügen).
3. Geben Sie die folgenden Eigenschaften an:

   Name

   Geben Sie dem Auslöser einen eindeutigen Namen.

   Auslösertyp

   Geben Sie eines der folgenden Elemente an:

- Schedule (Zeitplan): Der Auslöser wird mit einer bestimmten Häufigkeit und zu einem bestimmten Zeitpunkt ausgelöst.
- Job events (Jobereignisse): Ein bedingter Auslöser. Der Auslöser wird ausgelöst, wenn ein oder alle Aufträge in der Liste mit ihren designierten Zuständen übereinstimmen. Damit der Auslöser ausgelöst wird, müssen die überwachten Aufträge durch Auslöser gestartet worden sein. Für jeden ausgewählten Auftrag können Sie nur ein Auftragsereignis (Abschlusszustand) beobachten.
- On-demand (On-Demand): Der Auslöser wird ausgelöst, wenn er aktiviert wird.

4. Führen Sie den Auslöser-Assistenten aus. Auf der Seite Review (Überprüfen) können Sie durch Schedule (Zeitplan) und Job events (Auftragsereignisse) bedingte Auslöser sofort aktivieren, indem Sie Enable trigger on creation (Auslöser bei der Erstellung aktivieren) auswählen.

So fügen Sie einen Auslöser hinzu (AWS CLI)

- Verwenden Sie einen Befehl ähnlich dem folgenden.

```
aws glue create-trigger --name MyTrigger --type SCHEDULED --schedule  "cron(0 12 *
 * ? *)" --actions CrawlerName=MyCrawler --start-on-creation
```

Dieser Befehl erstellt einen Zeitplan-Auslöser mit dem Namen `MyTrigger`, der jeden Tag um 12:00 Uhr UTC ausgeführt wird und einen Crawler mit dem Namen `MyCrawler` startet. Der Auslöser wird im aktivierten Zustand erstellt.

Weitere Informationen finden Sie unter the section called “AWS Glue triggers”.

Zeitpläne für Aufträge und Crawler

Sie können einen Zeitplan für Ihre Crawler und Aufträge in AWS Glue definieren. Die Definition dieser Zeitpläne verwendet die Unix-ähnliche Cron-Syntax. Sie geben die Zeit in UTC (Coordinated Universal Time, koordinierte Weltzeit) an. Die minimale Genauigkeit für einen Zeitplan beträgt 5 Minuten.

Weitere Informationen zum Konfigurieren von Aufträgen und Crawlern für die Ausführung mit einem Zeitplan finden Sie unter Starten von Aufträgen und Crawlern über Auslöser.

Cron-Ausdrücke

Cron-Ausdrücke verfügen über sechs Pflichtfelder, die durch Leerzeichen voneinander getrennt sind.

Syntax

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Felder	Werte	Platzhalter
Minuten	0-59	, - * /
Stunden	0–23	, - * /
D ay-of-month	1-31	, - * ? / L W
Monat	1-12 oder JAN-DEZ	, - * /
D ay-of-week	1-7 oder SO-SA	, - * ? / L
Jahr	1970-2199	, - * /

Platzhalter

- Das Platzhalterzeichen , (Komma) schließt zusätzliche Werte ein. Im Feld `Month` würde `JAN,FEB,MAR` Januar, Februar und März abdecken.
- Das Platzhalterzeichen - (Bindestrich) gibt einen Bereich an. Im Feld `Day` steht 1-15 für die Tage 1 bis 15 des angegebenen Monats.
- Das Platzhalterzeichen * (Sternchen) steht für alle Werte im Feld. Im Feld `Hours` steht * für alle Stunden.
- Das Platzhalterzeichen / (Schrägstrich) steht für schrittweise Steigerungen. Im Feld `Minutes` können Sie **`1/10`** eingeben, um einen Bereich von je 10 Minuten beginnend mit der ersten Minute der Stunde anzugeben (z. B. die 11., 21. und 31. Minute usw.).
- Das Platzhalterzeichen ? (Fragezeichen) steht für einen Wert. In das `Day-of-month` Feld könntest du 7 eingeben, und wenn es dir egal wäre, welcher Wochentag der siebte war, könntest du eingeben? auf dem Day-of-week Feld.
- Das Platzhalterzeichen L in den Feldern für `Day-of-month` oder `Day-of-week` gibt den letzten Tag des Monats oder der Woche an.
- Das Platzhalterzeichen W im Feld `Day-of-month` gibt einen Wochentag an. Im Feld `Day-of-month` gibt den `3W` den Tag an, der dem dritten Tag des Monats am nächsten ist.

Einschränkungen

- Es ist nicht möglich, die Felder `Day-of-month` und `Day-of-week` im gleichen Cron-Ausdruck anzugeben. Wenn Sie einen Wert in einem der Felder angeben, müssen Sie in dem anderen Feld ein ? (Fragezeichen) eingeben.
- Cron-Ausdrücke, die zu schnelleren Häufigkeiten als mit 5 Minuten führen, werden nicht unterstützt.

Beispiele

Wenn Sie einen Zeitplan erstellen, können Sie die folgenden Beispiel-Cron-Strings verwenden.

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
0	10	*	*	?	*	Ausführung jeden Tag um 10:00 Uhr (UTC)
15	12	*	*	?	*	Ausführung jeden Tag um 12:15 Uhr (UTC)
0	18	?	*	MO-FR	*	Ausführung jeden Montag bis Freitag um 18:00 Uhr (UTC)
0	8	1	*	?	*	Ausführung jeden 1. Tag des Monats um

Minuten	Stunden	Tag des Monats	Monat	Wochentag	Jahr	Bedeutung
						08:00 Uhr (UTC)
0/15	*	*	*	?	*	Ausführung alle 15 Minuten
0/10	*	?	*	MO-FR	*	Ausführung alle 10 Minuten von Montag bis Freitag
0/5	8-17	?	*	MO-FR	*	Ausführung alle 5 Minuten von Montag bis Freitag zwischen 08:00 Uhr und 17:55 Uhr (UTC)

Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, geben Sie Folgendes an:

```
cron(15 12 * * ? *)
```

Aktivieren und Deaktivieren von Auslösern

Sie können einen Trigger aktivieren oder deaktivieren, indem Sie AWS Glue Konsole, die AWS Command Line Interface (AWS CLI) oder die AWS Glue API.

So aktivieren oder deaktivieren Sie einen Auslöser (Konsole)

1. Melden Sie sich an AWS Management Console und öffnen Sie das AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter ETL die Option Triggers (Auslöser) aus.
3. Aktivieren Sie das Kontrollkästchen neben dem gewünschten Auslöser und wählen Sie im Menü Action (Aktion) die Option Enable trigger (Auslöser aktivieren), um den Auslöser zu aktivieren, oder Disable trigger (Auslöser deaktivieren), um den Auslöser zu deaktivieren.

So aktivieren oder deaktivieren Sie einen Auslöser (AWS CLI)

- Geben Sie einen der folgenden Befehle ein.

```
aws glue start-trigger --name MyTrigger

aws glue stop-trigger --name MyTrigger
```

  Ein Auslöser wird beim Starten aktiviert und beim Stoppen deaktiviert. Wenn Sie einen On-Demand-Auslöser aktivieren, wird er sofort ausgelöst.

Weitere Informationen finden Sie unter the section called “AWS Glue triggers”.

Durchführung komplexer ETL-Aktivitäten mithilfe von Blueprints und Workflows in AWS Glue

Einige der komplexen ETL-Prozesse (Extrahieren, Transformieren und Laden) Ihres Unternehmens lassen sich am besten mithilfe mehrerer voneinander abhängiger Prozesse implementieren AWS Glue Jobs und Crawler. Die Verwendung von AWS Glue Workflows, Sie können einen komplexen ETL-Prozess mit mehreren Jobs und mehreren Crawlern entwerfen, der AWS Glue kann als eine Einheit ausgeführt und nachverfolgt werden. Nachdem Sie einen Workflow erstellt und die Aufträge, Crawler und Trigger im Workflow angegeben haben, können Sie den Workflow bei Bedarf oder nach einem Zeitplan ausführen.

Themen

- Überblick über Workflows in AWS Glue
- Manuelles Erstellen und Ausarbeiten eines Workflows in AWS Glue

- Einen AWS Glue Workflow mit einem EventBridge Amazon-Event starten
- EventBridge Ereignisse anzeigen, die einen Workflow gestartet haben
- Ausführen und Überwachen eines Workflows in AWS Glue
- Anhalten einer Workflow-Ausführung
- Reparieren und Fortsetzen einer Workflow-Ausführung
- Eigenschaften für Workflow-Ausführungen abrufen und festlegen in AWS Glue
- Abfragen von Workflows mit dem AWS Glue API
- Blueprint- und Workflow-Einschränkungen in AWS Glue
- Behebung von Blueprint-Fehlern in AWS Glue
- Berechtigungen für Personas und Rollen für AWS Glue-Blueprints

Überblick über Workflows in AWS Glue

In AWS Glue, können Sie Workflows verwenden, um komplexe ETL-Aktivitäten (Extrahieren, Transformieren und Laden) zu erstellen und zu visualisieren, an denen mehrere Crawler, Jobs und Trigger beteiligt sind. Jeder Workflow verwaltet die Ausführung und Überwachung aller zugehöriger Aufträge und Crawler. Wenn ein Workflow seine Komponenten ausführt, werden der Fortschritt und der Status der Ausführung aufgezeichnet. So erhalten Sie eine Übersicht über die Aufgabe und die Details der einzelnen Schritte. Das Tool AWS Glue Die Konsole bietet eine visuelle Darstellung eines Workflows als Diagramm.

Sie können einen Workflow aus einem erstellen AWS Glue Blueprint, oder Sie können einen Workflow manuell für eine Komponente nach der anderen erstellen, indem Sie den AWS Management Console oder den AWS Glue API verwenden. Weitere Informationen über Pläne finden Sie unter the section called “Übersicht über Blueprints”.

Auslöser innerhalb von Workflows können Aufträge und Crawler starten sowie von Aufträgen und Crawlern aktiviert werden. Mithilfe von Auslösern können Sie große Ketten voneinander abhängiger Aufträge und Crawler erstellen. Zusätzlich zu Auslösern in einem Workflow, die Auftrags- und Crawler-Abhängigkeiten definieren, verfügt jeder Workflow über einen Startauslöser. Es gibt drei Arten von Startauslösern:

- Zeitplan – Der Workflow wird entsprechend einem von Ihnen festgelegten Zeitplan gestartet. Der Start kann z. B. täglich, wöchentlich oder monatlich erfolgen, der Zeitplan kann aber auch benutzerdefiniert sein und auf einem `cron`-Ausdruck basieren.

- Bei Bedarf — Der Workflow wird manuell von der AWS Glue Konsole, API oder AWS CLI.
- EventBridge Ereignis — Der Workflow wird beim Eintreten eines einzelnen EventBridge Amazon-Ereignisses oder einer Reihe von EventBridge Amazon-Ereignissen gestartet. Mit diesem Triggertyp AWS Glue kann in einer ereignisgesteuerten Architektur ein Ereigniskonsument sein. Jeder EventBridge Ereignistyp kann einen Workflow starten. Ein häufiger Anwendungsfall ist die Ankunft eines neuen Objekts in einem Amazon-S3-Bucket (der S3-Vorgang `PutObject`).

  Der Start eines Workflows mit einem Batch von Ereignissen impliziert, dass gewartet wird, bis eine bestimmte Anzahl von Ereignissen empfangen wurde oder bis eine bestimmte Zeitspanne verstrichen ist. Wenn Sie den EventBridge Ereignisauslöser erstellen, können Sie optional Batch-Bedingungen angeben. Wenn Sie Batchbedingungen angeben, müssen Sie die Batchgröße (Anzahl der Ereignisse) und optional ein Batchfenster (Anzahl der Sekunden) angeben. Das standardmäßige und maximale Batchfenster beträgt 900 Sekunden (15 Minuten). Die Batchbedingung, die zuerst erfüllt ist, startet den Workflow. Das Batchfenster beginnt, wenn das erste Ereignis eintrifft. Wenn Sie beim Erstellen eines Auslösers keine Batchbedingungen angeben, wird die Batchgröße standardmäßig auf 1 gesetzt.

  Beim Start des Workflows werden die Batchbedingungen zurückgesetzt und der Ereignisauslöser beginnt zu überwachen, ob die nächste Batchbedingung für den erneuten Start des Workflows erfüllt ist.

  In der folgenden Tabelle wird gezeigt, wie Batchgröße und Batchfenster bei der Auslösung eines Workflows zusammenarbeiten.

Batchgröße	Batchfenster	Resultierende Auslösebedingung
10		Der Workflow wird beim Eintreffen von 10 EventBridge Ereignissen oder 15 Minuten nach dem Eintreffen des ersten Ereignisses ausgelöst, je nachdem, was zuerst eintritt. (Wenn die Fenstergröße nicht angegeben wurde, ist die Standardeinstellung 15 Minuten.)
10	2 Minuten	Der Workflow wird beim Eintreffen von 10 EventBridge Ereignissen oder 2 Minuten nach dem Eintreffen des ersten Ereignisses ausgelöst, je nachdem, was zuerst eintritt.

Batchgröße	Batchfenster	Resultierende Auslösebedingung
1		Der Workflow wird beim Eintreffen des ersten Ereignisses ausgelöst. Die Fenstergröße ist irrelevant. Die Batchgröße ist standardmäßig auf 1 festgelegt, wenn Sie bei der Erstellung des EventBridge Ereignisauslösers keine Batchbedingungen angeben.

Die `GetWorkflowRun`-API-Operation gibt die Batchbedingung zurück, die den Workflow ausgelöst hat.

Unabhängig davon, wie ein Workflow gestartet wird, können Sie beim Erstellen des Workflows die maximale Anzahl gleichzeitiger Workflow-Ausführungen angeben.

Wenn ein Ereignis oder ein Batch von Ereignissen eine Workflow-Ausführung startet, die fehlschlägt, wird dieses Ereignis oder dieser Batch von Ereignissen nicht mehr für den Start einer Workflow-Ausführung berücksichtigt. Eine neue Workflow-Ausführung wird erst gestartet, wenn das nächste Ereignis oder der nächste Batch von Ereignissen eintrifft.

> **Important**
>
> Beschränken Sie die Gesamtzahl der Aufträge, Crawler und Trigger innerhalb eines Workflows auf 100 oder weniger. Wenn Sie mehr als 100 einschließen, werden möglicherweise Fehler angezeigt, wenn Sie versuchen, Workflow-Läufe fortzusetzen oder zu beenden.

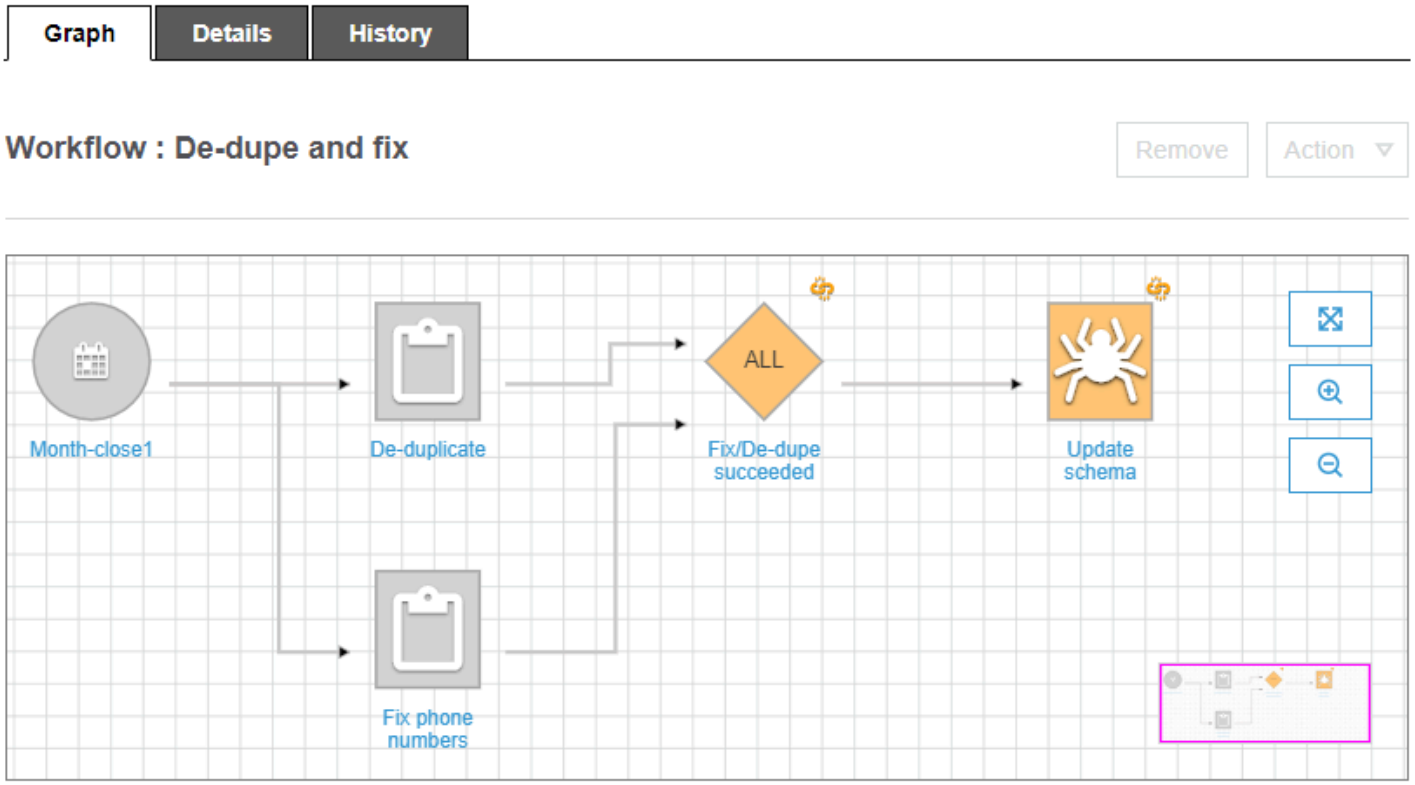
Eine Workflow-Ausführung wird nicht gestartet, wenn sie das für den Workflow festgelegte Parallelitätslimit überschreitet – auch wenn die Ereignisbedingung erfüllt ist. Es ist ratsam, die Grenzwerte für die Workflow-Parallelität an das erwartete Ereignisvolumen anzupassen. AWS Glue wiederholt keine Workflow-Läufe, die aufgrund einer Überschreitung der Parallelitätslimits fehlschlagen. Ebenso empfiehlt es sich, Parallelitätslimits für Aufträge und Crawler innerhalb von Workflows je nach dem erwarteten Ereignisvolumen anzupassen.

Eigenschaften von Workflow-Ausführungen

Zum Freigeben und Verwalten des Zustands während der Ausführung eines Workflows können Sie Standard-Ausführungseigenschaften für den Workflow definieren. Bei diesen Eigenschaften handelt es sich um Name/Wert-Paare, die für alle Aufträge im Workflow verfügbar sind. Mithilfe von können Jobs die AWS Glue API Eigenschaften der Workflow-Ausführung abrufen und sie für Jobs ändern, die später im Workflow erscheinen.

Workflow-Diagramm

Die folgende Abbildung zeigt das Diagramm eines sehr einfachen Workflows auf der AWS Glue console. Ein Workflow kann Dutzende von Komponenten umfassen.



Dieser Workflow wird von einem Zeitplanauslöser (`Month-close1`) gestartet, der zwei Aufträge startet, `De-duplicate` und `Fix phone numbers`. Nach erfolgreichem Abschluss beider Aufträge startet ein Ereignisauslöser (`Fix/De-dupe succeeded`) einen Crawler (`Update schema`).

Statische und dynamische Workflow-Ansichten

Für jeden Workflow gibt es eine statische und eine dynamische Ansicht. Die statische Ansicht zeigt die Struktur des Workflows. Die dynamische Ansicht ist eine Laufzeitansicht, die die aktuellen Ausführungsinformationen für alle Aufträge und Crawler enthält. Zu Ausführungsinformationen gehören zu Erfolgsstatus und Fehlerdetails.

Während ein Workflow ausgeführt wird, wird in der Konsole die dynamische Ansicht angezeigt. Diese stellt die abgeschlossenen und noch ausstehenden Aufträge grafisch dar. Sie können auch mit der AWS Glue API eine dynamische Ansicht eines aktiven Workflows abrufen. Weitere Informationen finden Sie unter Abfragen von Workflows mit dem AWS Glue API.

Weitere Informationen finden Sie auch unter

- the section called “Erstellen eines Workflows aus einem Blueprint”
- the section called “Manuelles Erstellen und Aufbauen eines Workflows”
- Workflows (für die Workflow-API)

Manuelles Erstellen und Ausarbeiten eines Workflows in AWS Glue

Sie können das AWS Glue Konsole, um einen Workflow knotenweise manuell zu erstellen und auszuarbeiten.

Ein Workflow enthält Aufträge, Crawler und Auslöser. Erstellen Sie vor dem manuellen Erstellen eines Workflows die Aufträge und Crawler, die der Workflow enthalten soll. Es empfiehlt sich, run-on-demand Crawler für Workflows anzugeben. Sie können neue Auslöser erstellen, während Sie Ihren Workflow erstellen, oder Sie können vorhandene Auslöser in den Workflow klonen. Wenn Sie einen Auslöser klonen, werden alle dem Auslöser zugeordneten Katalogobjekte – die Aufträge oder Crawler, die ihn auslösen, sowie die Aufträge oder Crawler, die er startet, dem Workflow hinzugefügt.

Important

Beschränken Sie die Gesamtzahl der Aufträge, Crawler und Trigger innerhalb eines Workflows auf 100 oder weniger. Wenn Sie mehr als 100 einschließen, werden möglicherweise Fehler angezeigt, wenn Sie versuchen, Workflow-Läufe fortzusetzen oder zu beenden.

Der Workflow wird weiter bearbeitet, indem Sie dem Workflow-Diagramm Auslöser hinzufügen und die überwachten Ereignisse und Aktionen für jeden Auslöser definieren. Sie beginnen mit einem Startauslöser. Dabei kann es sich um einen On-Demand-Auslöser oder einen Zeitplanauslöser handeln. Dann schließen Sie das Diagramm ab, indem Sie Ereignisauslöser (bedingte Auslöser) hinzufügen.

Schritt 1: Erstellen des Workflows

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter ETL die Option Workflows aus.
3. Wählen Sie Add Workflow (Workflow hinzufügen) aus und füllen Sie das Formular Add a new ETL workflow (Neuen ETL-Workflow hinzufügen) aus.

   Alle optionalen Standard-Ausführungseigenschaften, die Sie hinzufügen, werden als Argumente für alle Aufträge im Workflow verfügbar gemacht. Weitere Informationen finden Sie unter Eigenschaften für Workflow-Ausführungen abrufen und festlegen in AWS Glue.
4. Wählen Sie Add Workflow (Workflow hinzufügen) aus.

   Der neue Workflow wird in der Liste auf der Seite Workflows angezeigt.

Schritt 2: Hinzufügen eines Startauslösers

1. Wählen Sie auf der Seite Workflows den neuen Workflow aus. Wählen Sie unten auf der Seite die Registerkarte Graph (Diagramm) aus.
2. Wählen Sie Add trigger (Auslöser hinzufügen) aus und führen Sie im Dialogfeld Add trigger (Auslöser hinzufügen) einen der folgenden Schritte aus:

   - Wählen Sie Clone existing (Vorhandenen klonen) aus und wählen Sie einen vorhandenen Auslöser aus, den Sie klonen möchten. Wählen Sie dann Add (Hinzufügen).

     Der Auslöser wird im Diagramm angezeigt, zusammen mit den Aufträgen und Crawlern, die er überwacht, sowie den Aufträgen und Crawlern, die er startet.

     Wenn Sie versehentlich den falschen Auslöser ausgewählt haben, wählen Sie den Auslöser im Diagramm aus und klicken Sie dann auf Remove (Entfernen).
   - Wählen Sie Add new (Neu) aus und füllen Sie das Formular Add trigger (Auslöser hinzufügen) aus.

     1. Wählen Sie als Triggertyp die Option Zeitplan, Auf Abruf oder EventBridgeEreignis aus.

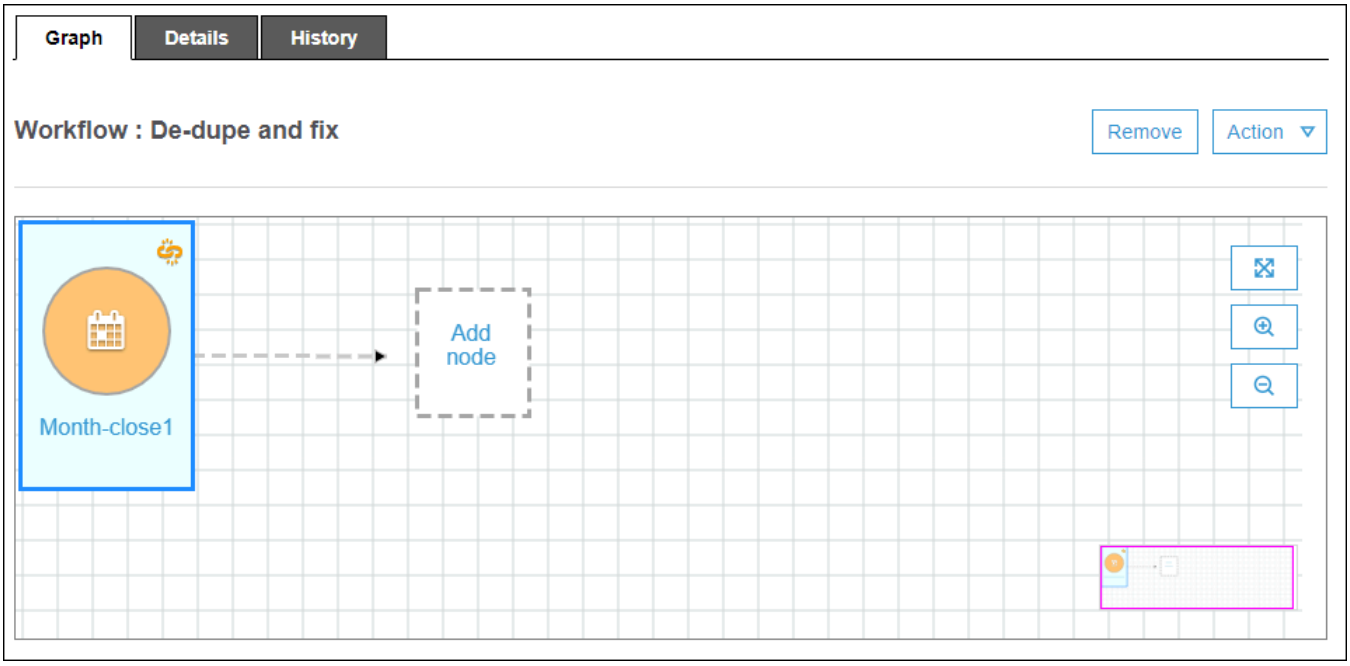
        Wählen Sie für den Auslösertyp Schedule (Zeitplan) eine der Optionen für Frequency (Häufigkeit) aus. Wählen Sie Custom (Benutzerdefiniert) aus und geben Sie einen `cron`-Ausdruck ein.

Geben Sie als EventBridge Ereignis vom Typ Trigger die Anzahl der Ereignisse (Stapelgröße) und optional Zeitverzögerung (Batchfenster) ein. Wenn Sie keinen Wert für Time delay (Verzögerung) angeben, wird das Batchzeitfenster standardmäßig auf 15 Minuten eingestellt. Weitere Informationen finden Sie unter Überblick über Workflows in AWS Glue.

2. Wählen Sie Hinzufügen aus.

Der Auslöser wird im Diagramm mit einem Platzhalterknoten angezeigt. Dieser ist mit Add node (Knoten hinzufügen) markiert. Im folgenden Beispiel ist der Startauslöser ein Zeitplanauslöser namens `Month-close1`.

Bis jetzt wurde der Auslöser noch nicht gespeichert.



3. Wenn Sie einen neuen Auslöser hinzugefügt haben, führen Sie die folgenden Schritte aus:

   a. Führen Sie eine der folgenden Aktionen aus:

   - Wählen Sie den Platzhalterknoten mit der Bezeichnung Add node (Knoten hinzufügen) aus.
   - Stellen Sie sicher, dass der Auslöser ausgewählt ist, und wählen Sie über dem Diagramm im Menü Action (Aktion) die Option Add jobs/crawler to trigger (Aufträge/Crawler zu Auslöser hinzufügen) aus.

b. Wählen Sie im Dialogfeld Add jobs(s) and crawler(s) to trigger (Aufträge/Crawler zu Auslöser hinzufügen) mindestens einen Auftrag oder Crawler aus. Klicken Sie dann auf Add (Hinzufügen).

   Der Auslöser wird gespeichert und die ausgewählten Aufträge oder Crawler werden im Diagramm mit Verbindungen zum Auslöser angezeigt.

   Wenn Sie versehentlich die falschen Aufträge oder Crawler hinzugefügt haben, können Sie den Auslöser oder eine Verbindung auswählen und auf Remove (Entfernen) klicken.

Schritt 3: Hinzufügen weiterer Auslöser

Sie können den Workflow erweitern und weitere Auslöser vom Typ Event (Ereignis) hinzufügen. Mit den Symbolen rechts neben dem Diagramm können Sie die Zeichenfläche vergrößern oder verkleinern. Führen Sie für jeden Auslöser, den Sie hinzufügen möchten, die folgenden Schritte aus:

> **Note**
>
> Es gibt keine Aktion, um den Workflow zu speichern. Nachdem Sie den letzten Auslöser hinzugefügt und ihm Aktionen zugewiesen haben, wird der Workflow abgeschlossen und gespeichert. Sie können jederzeit später weitere Knoten hinzufügen.

1. Führen Sie eine der folgenden Aktionen aus:

   - Wenn Sie einen vorhandenen Auslöser klonen möchten, stellen Sie sicher, dass kein Knoten im Diagramm ausgewählt ist. Wählen Sie dann im Menü Action (Aktion) die Option Add trigger (Auslöser hinzufügen) aus.
   - Wenn Sie einen neuen Auslöser hinzufügen möchten, der einen bestimmten Auftrag oder Crawler im Diagramm überwacht, wählen Sie den Auftrags- oder Crawler-Knoten aus. Wählen Sie dann den Platzhalterknoten Add trigger (Auslöser hinzufügen) aus.

     In einem späteren Schritt können Sie weitere Aufträge oder Crawler hinzufügen, die diesen Auslöser überwachen.

2. Führen Sie im Dialogfeld Add trigger (Auslöser hinzufügen) einen der folgenden Schritte aus:

   - Wählen Sie Add new (Neu) aus und füllen Sie das Formular Add trigger (Auslöser hinzufügen) aus. Wählen Sie dann Add (Hinzufügen).

Der Auslöser wird im Diagramm dargestellt. Sie stellen den Auslöser in einem späteren Schritt fertig.

- Wählen Sie Clone existing (Vorhandenen klonen) aus und wählen Sie einen vorhandenen Auslöser aus, den Sie klonen möchten. Wählen Sie dann Add (Hinzufügen).

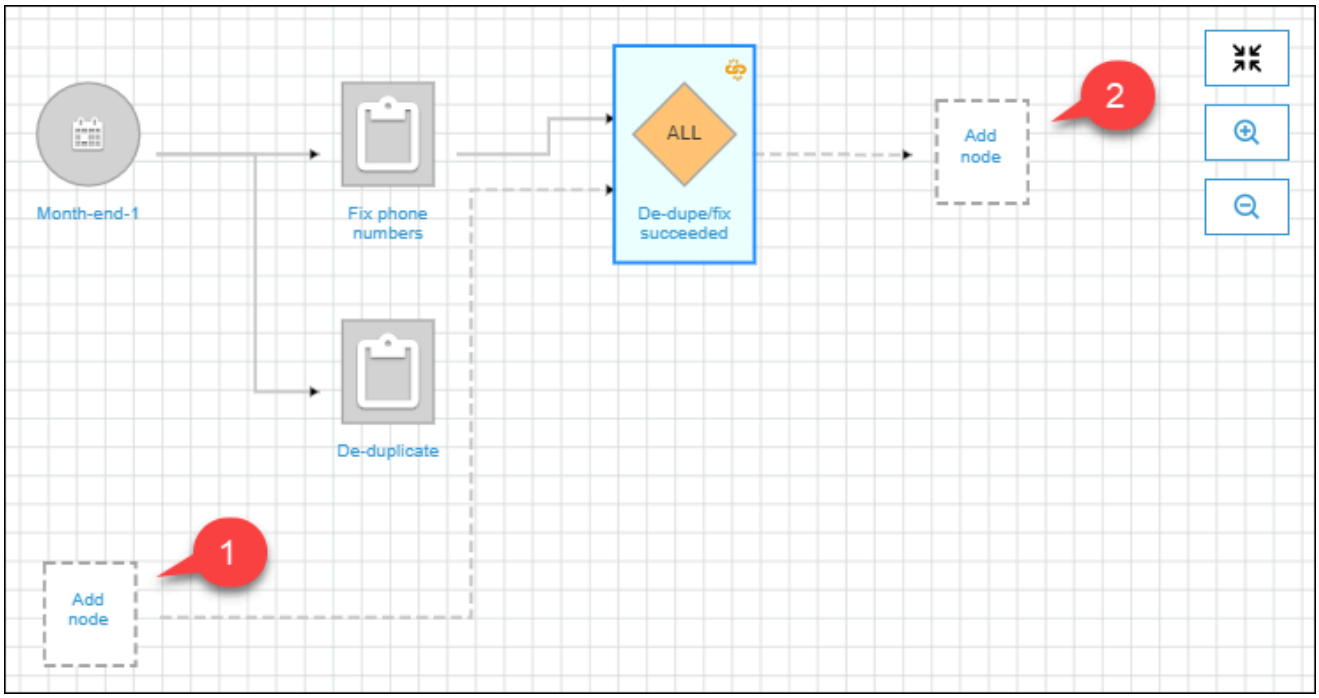
  Der Auslöser wird im Diagramm angezeigt, zusammen mit den Aufträgen und Crawlern, die er überwacht, sowie den Aufträgen und Crawlern, die er startet.

  Wenn Sie versehentlich den falschen Auslöser ausgewählt haben, markieren Sie ihn im Diagramm und klicken Sie dann auf Remove (Entfernen).

3. Wenn Sie einen neuen Auslöser hinzugefügt haben, führen Sie die folgenden Schritte aus:

   a. Wählen Sie den neuen Trigger aus.

      Der Auslöser `De-dupe/fix succeeded` ist ausgewählt und für zu überwachende Ereignisse (1) und Aktionen (2) werden Platzhalterknoten angezeigt, wie in der folgenden Abbildung dargestellt.



   b. (Optional, wenn der Auslöser bereits ein Ereignis überwacht und Sie weitere zu überwachende Aufträge oder Crawler hinzufügen möchten.) Wählen Sie den events-to-watch Platzhalterknoten und wählen Sie im Dialogfeld Zu überwachende Jobs und Crawler hinzufügen einen oder mehrere Jobs oder Crawler aus. Wählen Sie ein zu überwachendes Ereignis aus (SUCCEEDED, FAILED usw.) und klicken Sie auf Add (Hinzufügen).

c. Stellen Sie sicher, dass der Auslöser ausgewählt ist, und wählen Sie den Platzhalterknoten für Aktionen aus.

d. Wählen Sie im Dialogfeld Add job(s) and crawler(s) to watch (Zu überwachende Aufträge und Crawler hinzufügen) mindestens einen Auftrag oder Crawler aus. Klicken Sie dann auf Add (Hinzufügen).

   Die ausgewählten Aufträge und Crawler werden im Diagramm mit Verbindungen zum Auslöser angezeigt.

Weitere Informationen zu Workflows und Blueprints finden Sie in den folgenden Themen.

- Überblick über Workflows in AWS Glue
- Ausführen und Überwachen eines Workflows in AWS Glue
- Erstellen eines Workflows aus einem Blueprint in AWS Glue

Einen AWS Glue Workflow mit einem EventBridge Amazon-Event starten

Amazon EventBridge, auch bekannt als CloudWatch Events, ermöglicht es Ihnen, Ihre AWS Services zu automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen zu reagieren. Ereignisse im AWS Rahmen von Services werden EventBridge nahezu in Echtzeit übermittelt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen durchgeführt werden sollen, wenn sich für ein Ereignis eine Übereinstimmung mit einer Regel ergibt.

Mit EventBridge Support AWS Glue kann er in einer ereignisgesteuerten Architektur als Produzent und Kunde von Veranstaltungen agieren. AWS GlueUnterstützt bei Workflows jede Art von EventBridge Ereignis als Verbraucher. Der wahrscheinlich häufigste Anwendungsfall ist die Ankunft eines neuen Objekts in einem Amazon-S3-Bucket. Wenn Daten in unregelmäßigen oder undefinierten Intervallen eintreffen, können Sie diese Daten so zeitnah wie möglich verarbeiten.

> **Note**
>
> AWS Gluebietet keine garantierte Zustellung von EventBridge Nachrichten. AWS Glueführt keine Deduplizierung durch, wenn doppelte Nachrichten EventBridge zugestellt werden. Sie müssen die Idempotenz auf der Grundlage Ihres Anwendungsfalls verwalten.

Achten Sie darauf, die EventBridge Regeln korrekt zu konfigurieren, um das Senden unerwünschter Ereignisse zu vermeiden.

Bevor Sie beginnen

Wenn Sie einen Workflow mit Amazon S3 S3-Datenereignissen starten möchten, müssen Sie sicherstellen, dass Ereignisse für den gewünschten S3-Bucket in AWS CloudTrail und protokolliert werden EventBridge. Dazu müssen Sie einen CloudTrail Trail erstellen. Weitere Informationen findest du unter Einen Trail für dein AWS Konto erstellen.

Um einen Workflow mit einem EventBridge Ereignis zu starten

Note

Ersetzen Sie in den folgenden Befehlen …

- *<workflow-name>*mit dem Namen, der dem Workflow zugewiesen werden soll.
- *<trigger-name>*mit dem Namen, der dem Trigger zugewiesen werden soll.
- *<bucket-name>*mit dem Namen des Amazon S3 S3-Buckets.
- *<account-id>*mit einer gültigen AWS Konto-ID.
- *<region>*mit dem Namen der Region (zum Beispiel`us-east-1`).
- *<rule-name>*mit dem Namen, der der EventBridge Regel zugewiesen werden soll.

1. Stellen Sie sicher, dass Sie über AWS Identity and Access Management (IAM-) Berechtigungen zum Erstellen und Anzeigen von EventBridge Regeln und Zielen verfügen. Nachfolgend finden Sie eine Musterrichtlinie, die Sie anfügen können. Möglicherweise sollten Sie sie eingrenzen, um Beschränkungen für die Vorgänge und Ressourcen festzulegen.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "events:PutRule",
                "events:DisableRule",
                "events:DeleteRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "events:EnableRule",
                "events:List*",
                "events:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Erstellen Sie eine IAM-Rolle, die der EventBridge Dienst übernehmen kann, wenn er ein Ereignis weiterleitet. AWS Glue

   a. Wählen Sie auf der Seite Create role (Rolle erstellen) der IAM-Konsole die Option AWS - Service aus. Wählen Sie dann den Service CloudWatch Events aus.

   b. Schließen Sie den Assistenten Create role (Rolle erstellen) ab. Der Assistent fügt automatisch die Richtlinien `CloudWatchEventsBuiltInTargetExecutionAccess` und `CloudWatchEventsInvocationAccess` an.

   c. Fügen Sie der Rolle die folgende Inline-Richtlinie an. Diese Richtlinie ermöglicht es dem EventBridge Dienst, Ereignisse an sie weiterzuleitenAWS Glue.

      JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:notifyEvent"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<account-id>:workflow/<workflow-
name>"
            ]
        }
    ]
```

```
}
```

3. Verwenden Sie den folgenden Befehl, um den Workflow zu erstellen.

   Weitere Informationen zu zusätzlichen optionalen Befehlszeilenparametern finden Sie unter create-workflow in der AWS CLI -Befehlsreferenz.

   ```
   aws glue create-workflow --name <workflow-name>
   ```

4. Geben Sie den folgenden Befehl ein, um einen EventBridge Ereignisauslöser für den Workflow zu erstellen. Dies wird der Startauslöser für den Workflow sein. *<actions>*Ersetzen Sie ihn durch die auszuführenden Aktionen (die Jobs und Crawler, die gestartet werden sollen).

   Weitere Informationen zum Programmieren des `actions`-Arguments finden Sie unter create-trigger in der AWS CLI -Befehlsreferenz.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT --
   name <trigger-name> --actions <actions>
   ```

   Wenn Sie möchten, dass der Workflow durch eine Reihe von Ereignissen statt durch ein einzelnes EventBridge Ereignis ausgelöst wird, geben Sie stattdessen den folgenden Befehl ein.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT
    --name <trigger-name> --event-batching-condition BatchSize=<number-of-
   events>,BatchWindow=<seconds> --actions <actions>
   ```

   Für das `event-batching-condition`-Argument ist `BatchSize` erforderlich und `BatchWindow` optional. Wenn Sie `BatchWindow` weglassen, beträgt das Fenster standardmäßig 900 Sekunden – die maximale Fenstergröße.

   Example

   Im folgenden Beispiel wird ein Trigger erstellt, der den `eventtest` Workflow startet, nachdem drei EventBridge Ereignisse eingetroffen sind, oder fünf Minuten nach dem Eintreffen des ersten Ereignisses, je nachdem, was zuerst eintritt.

   ```
   aws glue create-trigger --workflow-name eventtest --type EVENT --name objectArrival
    --event-batching-condition BatchSize=3,BatchWindow=300 --actions JobName=test1
   ```

5. Erstellen Sie eine Regel in Amazon EventBridge.

a. Erstellen Sie das JSON-Objekt für die Regeldetails in Ihrem bevorzugten Texteditor.

Im folgenden Beispiel wird Amazon S3 als Ereignisquelle, `PutObject` als Ereignisname und der Bucket-Name als Anforderungsparameter angegeben. Diese Regel startet einen Workflow, wenn ein neues Objekt im Bucket eintrifft.

```
{
  "source": [
    "aws.s3"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3.amazonaws.com"
    ],
    "eventName": [
      "PutObject"
    ],
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
    }
  }
}
```

Damit der Workflow gestartet wird, wenn ein neues Objekt in einem Ordner innerhalb des Buckets eintrifft, können Sie den folgenden Code für `requestParameters` austauschen.

```
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
      "key" : [{ "prefix" : "<folder1>/<folder2>/*"}}]
  }
```

b. Verwenden Sie Ihr bevorzugtes Tool, um das JSON-Regelobjekt in eine Zeichenfolge mit Escapezeichen zu konvertieren.

```
{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-type\": [\n    \"AWS API
 Call via CloudTrail\"\n  ],\n  \"detail\": {\n    \"eventSource\": [\n
 \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n      \"PutObject\"\n
  ],\n    \"requestParameters\": {\n      \"bucketName\": [\n        \"<bucket-
name>\"\n      ]\n    }\n  }\n}
```

c. Führen Sie den folgenden Befehl aus, um eine JSON-Parametervorlage zu erstellen, die Sie bearbeiten können, um Eingabeparameter für einen nachfolgenden `put-rule`-Befehl anzugeben. Speichern Sie die Ausgabe in einer Datei. In diesem Beispiel heißt die Datei `ruleCommand`.

```
aws events put-rule --name <rule-name> --generate-cli-skeleton >ruleCommand
```

Weitere Informationen zum Parameter `--generate-cli-skeleton` finden Sie unter Generieren der AWS CLI -Skeleton- und -Eingabeparameter aus einer JSON- oder YAML-Eingabedatei im Benutzerhandbuch für die AWS -Befehlszeilenschnittstelle.

Die Ausgabedatei sollte wie folgt aussehen.

```
{
    "Name": "",
    "ScheduleExpression": "",
    "EventPattern": "",
    "State": "ENABLED",
    "Description": "",
    "RoleArn": "",
    "Tags": [
        {
            "Key": "",
            "Value": ""
        }
    ],
    "EventBusName": ""
}
```

d. Bearbeiten Sie die Datei, um mindestens die Parameter `Name`, `EventPattern` und `State` festzulegen und optional Parameter zu entfernen. Geben Sie für den Parameter `EventPattern` die Zeichenfolge mit Escapezeichen für die Regeldetails an, die Sie in einem vorherigen Schritt erstellt haben.

```
{
    "Name": "<rule-name>",
    "EventPattern": "{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-
type\": [\n    \"AWS API Call via CloudTrail\"\n  ],\n  \"detail\": {\n
 \"eventSource\": [\n      \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n
      \"PutObject\"\n    ],\n    \"requestParameters\": {\n      \"bucketName
\": [\n        \"<bucket-name>\"\n      ]\n    }\n  }\n}",
    "State": "DISABLED",
    "Description": "Start an AWS Glue workflow upon new file arrival in an
 Amazon S3 bucket"
}
```

> **Note**
>
> Es empfiehlt sich, die Regel deaktiviert zu lassen, bis Sie den Workflow fertig erstellt haben.

e. Geben Sie den folgenden `put-rule`-Befehl ein, der Eingabeparameter aus der Datei `ruleCommand` liest.

```
aws events put-rule --name <rule-name> --cli-input-json file://ruleCommand
```

Die folgende Ausgabe zeigt den Erfolg an.

```
{
    "RuleArn": "<rule-arn>"
}
```

6. Führen Sie den folgenden Befehl aus, um die Richtlinie einem Ziel anzufügen. Das Ziel ist der Workflow in AWS Glue. *<role-name>*Ersetzen Sie sie durch die Rolle, die Sie zu Beginn dieses Verfahrens erstellt haben.

```
aws events put-targets --rule <rule-name> --targets
 "Id"="1","Arn"="arn:aws:glue:<region>:<account-id>:workflow/<workflow-
name>","RoleArn"="arn:aws:iam::<account-id>:role/<role-name>" --region <region>
```

Die folgende Ausgabe zeigt den Erfolg an.

```
{
```

```
    "FailedEntryCount": 0,
    "FailedEntries": []
}
```

7. Bestätigen Sie die erfolgreiche Verbindung von Regel und Ziel, indem Sie den folgenden Befehl eingeben.

```
aws events list-rule-names-by-target --target-arn arn:aws:glue:<region>:<account-id>:workflow/<workflow-name>
```

Die folgende Ausgabe weist auf Erfolg hin. Dabei *<rule-name>* steht der Name der Regel, die Sie erstellt haben.

```
{
    "RuleNames": [
        "<rule-name>"
    ]
}
```

8. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
9. Wählen Sie den Workflow aus und prüfen Sie, ob der Startauslöser und seine Aktionen – die zu startenden Aufträge oder Crawler – im Workflow-Diagramm angezeigt werden. Fahren Sie anschließend mit dem Verfahren in Schritt 3: Hinzufügen weiterer Auslöser fort. Oder fügen Sie dem Workflow über die AWS Glue-API oder - AWS Command Line Interface weitere Komponenten hinzu.
10. Wenn der Workflow vollständig angegeben ist, aktivieren Sie die Regel.

```
aws events enable-rule --name <rule-name>
```

Der Workflow kann jetzt durch ein EventBridge Ereignis oder einen Ereignisstapel gestartet werden.

Weitere Informationen finden Sie auch unter

- EventBridge Amazon-Benutzerhandbuch
- Überblick über Workflows in AWS Glue

- Manuelles Erstellen und Ausarbeiten eines Workflows in AWS Glue

EventBridge Ereignisse anzeigen, die einen Workflow gestartet haben

Sie können die Event-ID des EventBridge Amazon-Events einsehen, mit dem Ihr Workflow gestartet wurde. Wenn Ihr Workflow durch einen Stapel von Ereignissen gestartet wurde, können Sie das Ereignis IDs aller Ereignisse im Batch anzeigen.

Bei Workflows, deren Batchgröße größer als eins ist, können Sie außerdem sehen, welche Batchbedingung den Workflow gestartet hat: die Ankunft der Anzahl der Ereignisse in der Batchgröße oder der Ablauf des Batchfensters.

Um die EventBridge Ereignisse anzuzeigen, die einen Workflow gestartet haben (Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option Workflows aus.
3. Wählen Sie den Workflow aus. Wählen Sie unten die Registerkarte History (Verlauf) aus.
4. Wählen Sie eine Workflow-Ausführung und dann die Option View run details (Ausführungsdetails anzeigen) aus.
5. Suchen Sie auf der Seite mit den Ausführungsdetails das Feld Run properties (Ausführungseigenschaften) und dann den Schlüssel aws:eventIds.

   Der Wert für diesen Schlüssel ist eine Liste von EventBridge Ereignissen IDs.

Um die EventBridge Ereignisse anzuzeigen, die einen Workflow (AWS API) gestartet haben

- Fügen Sie den folgenden Code in Ihr Python-Skript ein.

```
workflow_params =
 glue_client.get_workflow_run_properties(Name=workflow_name,RunId=workflow_run_id)
batched_events = workflow_params['aws:eventIds']
```

  `batched_events` sollte eine Liste von Zeichenfolgen sein, die jeweils eine Ereignis-ID darstellen.

Weitere Informationen finden Sie auch unter

- EventBridge Amazon-Benutzerhandbuch
- the section called “Übersicht über Workflows”

Ausführen und Überwachen eines Workflows in AWS Glue

Wenn es sich bei dem Start-Trigger für einen Workflow um einen On-Demand-Trigger handelt, können Sie den Workflow über den AWS Glue console. Führen Sie die folgenden Schritte aus, um einen Workflow auszuführen und zu überwachen. Wenn der Workflow fehlschlägt, können Sie das Ausführungsdiagramm aufrufen, um den fehlgeschlagenen Knoten zu ermitteln. Wenn der Workflow mit einem Blueprint erstellt wurde, können Sie die Blueprint-Ausführung aufrufen, um die Blueprint-Parameterwerte anzuzeigen, die zum Erstellen des Workflows verwendet wurden. Weitere Informationen finden Sie unter the section called “Anzeigen von Blueprint-Ausführungen”.

Sie können einen Workflow ausführen und überwachen, indem Sie AWS Glue Konsole, API oder AWS Command Line Interface (AWS CLI).

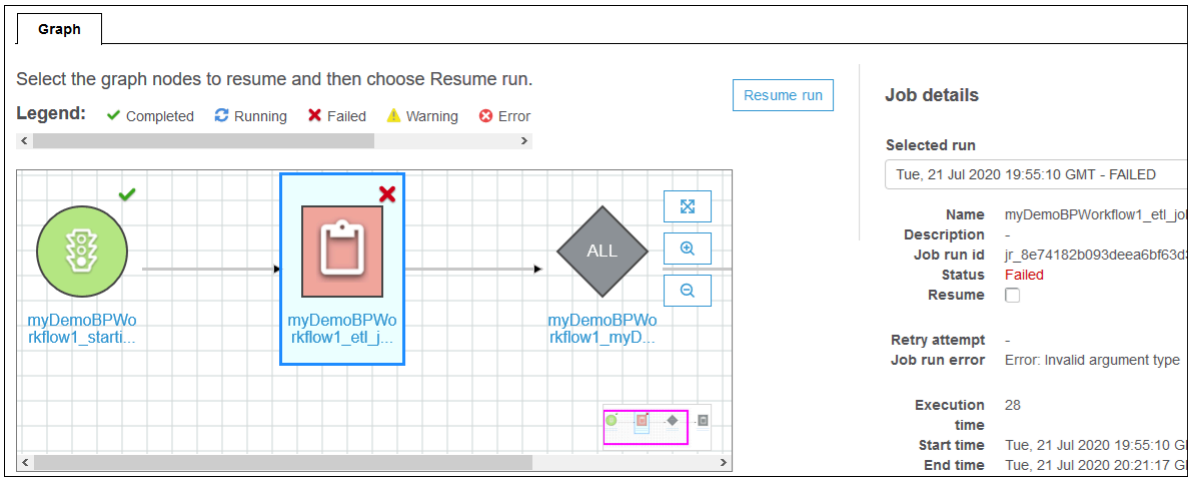
Einen Workflow ausführen und überwachen (Konsole)

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter ETL die Option Workflows aus.
3. Wählen Sie einen Workflow aus. Wählen Sie im Menü Actions (Aktionen) die Option Edit (Bearbeiten) aus.
4. Überprüfen Sie die Spalte Last run status (Status der letzten Ausführung) in der Workflow-Liste. Wählen Sie die Schaltfläche „Refresh“ (Aktualisieren) aus, um den Status des laufenden Workflows anzuzeigen.
5. Während der Workflow ausgeführt wird oder nachdem er abgeschlossen wurde (oder fehlgeschlagen ist), können Sie die Ausführungsdetails abrufen. Dazu führen Sie die folgenden Schritte aus.

    a. Wählen Sie den Workflow aus und gehen Sie auf die Registerkarte History (Verlauf).
    b. Wählen Sie die aktuelle oder letzte Workflow-Ausführung und dann View run details (Ausführungsdetails anzeigen) aus.

    Das Workflow-Laufzeitdiagramm zeigt den aktuellen Ausführungsstatus an.

c. Wählen Sie einen beliebigen Knoten im Diagramm aus, um Details und Status des Knotens anzuzeigen.



Einen Workflow ausführen und überwachen (AWS CLI)

1. Geben Sie den folgenden Befehl ein. *<workflow-name>*Ersetzen Sie es durch den auszuführenden Workflow.

```
aws glue start-workflow-run --name <workflow-name>
```

Wenn der Workflow erfolgreich gestartet wurde, gibt der Befehl die Ausführungs-ID zurück.

2. Rufen Sie mithilfe des Befehls `get-workflow-run` den Status der Workflow-Ausführung ab. Geben Sie den Namen und die Ausführungs-ID des Workflows an.

```
aws glue get-workflow-run --name myWorkflow --run-id
 wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705
```

Hier sehen Sie eine beispielhafte Befehlsausgabe.

```
{
    "Run": {
        "Name": "myWorkflow",
        "WorkflowRunId":
 "wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705",
        "WorkflowRunProperties": {
            "run_state": "COMPLETED",
            "unique_id": "fee63f30-c512-4742-a9b1-7c8183bdaae2"
```

```
        },
        "StartedOn": 1578556843.049,
        "CompletedOn": 1578558649.928,
        "Status": "COMPLETED",
        "Statistics": {
            "TotalActions": 11,
            "TimeoutActions": 0,
            "FailedActions": 0,
            "StoppedActions": 0,
            "SucceededActions": 9,
            "RunningActions": 0,
            "ErroredActions": 0
        }
    }
}
```

Weitere Informationen finden Sie auch unter:

- the section called “Übersicht über Workflows”
- the section called “Übersicht über Blueprints”

Anhalten einer Workflow-Ausführung

Sie können das AWS Glue Konsole, AWS Command Line Interface (AWS CLI) oder AWS Glue API zum Beenden einer Workflow-Ausführung. Wenn Sie einen Workflow anhalten, werden alle laufenden Aufgaben und Crawler sofort beendet und Aufgaben und Crawler, die noch nicht gestartet wurden, werden nie gestartet. Es kann bis zu einer Minute dauern, bis alle laufenden Aufgaben und Crawler beendet sind. Der Workflow-Ausführungsstatus wechselt von Running (Wird ausgeführt) zu Stopping (Wird angehalten), und wenn die Workflow-Ausführung vollständig angehalten ist, wechselt der Status zu Stopped (Angehalten).

Nachdem die Workflowlausführung angehalten wurde, können Sie das Ausführungsdiagramm anzeigen, um zu sehen, welche Aufgaben und Crawler abgeschlossen wurden und welche nie gestartet wurden. Anschließend können Sie ermitteln, ob Sie Schritte ausführen müssen, um die Datenintegrität zu gewährleisten. Das Anhalten einer Workflow-Ausführung führt dazu, dass keine automatischen Rollback-Operationen ausgeführt werden.

So beenden Sie eine Workflow-Ausführung (Konsole):

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich unter ETL die Option Workflows aus.
3. Wählen Sie einen derzeit ausgeführten Workflow und dann die Registerkarte History (Verlauf) aus.
4. Wählen Sie die Workflow-Ausführung aus und klicken Sie dann auf Stop run (Ausführen beenden).

   Der Ausführungsstatus ändert sich zu Stopping (Wird angehalten).
5. (Optional) Wählen Sie die Workflow-Ausführung aus, wählen Sie View run details (Ausführungsdetails anzeigen) und überprüfen Sie das Ausführungsdiagramm.

So beenden Sie eine Workflow-Ausführung (AWS CLI):

- Geben Sie den folgenden Befehl ein. *<workflow-name>*Ersetzen Sie es durch den Namen des Workflows und *<run-id>* durch die Ausführungs-ID des Workflows, der beendet werden soll.

  ```
  aws glue stop-workflow-run --name <workflow-name> --run-id <run-id>
  ```

  Im Folgenden finden Sie ein Beispiel für den stop-workflow-run-Befehl.

  ```
  aws glue stop-workflow-run --name my-workflow --run-id
   wr_137b88917411d128081069901e4a80595d97f719282094b7f271d09576770354
  ```

Reparieren und Fortsetzen einer Workflow-Ausführung

Wenn ein oder mehrere Knoten (Aufträge oder Crawler) in einem Workflow nicht erfolgreich abgeschlossen werden, wurde der Workflow nur teilweise ausgeführt. Nachdem Sie die Ursachen ermittelt und Korrekturen vorgenommen haben, können Sie einen oder mehrere Knoten auswählen, von denen die Workflow-Ausführung fortgesetzt werden soll, und die Ausführung dann fortsetzen. Dadurch werden die ausgewählten Knoten und alle nachgelagerten Knoten ausgeführt.

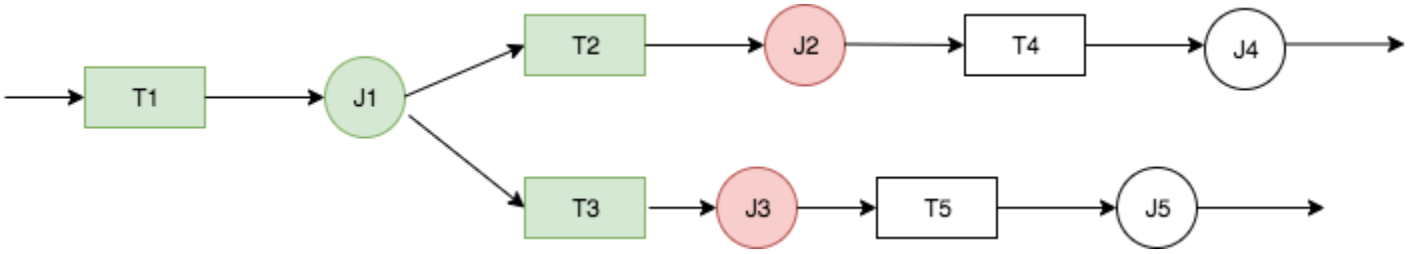
Themen

- Fortsetzen einer Workflow-Ausführung: Funktionsweise
- Fortsetzen einer Workflow-Ausführung

- Hinweise und Einschränkungen für die Fortsetzung von Workflow-Ausführungen

Fortsetzen einer Workflow-Ausführung: Funktionsweise

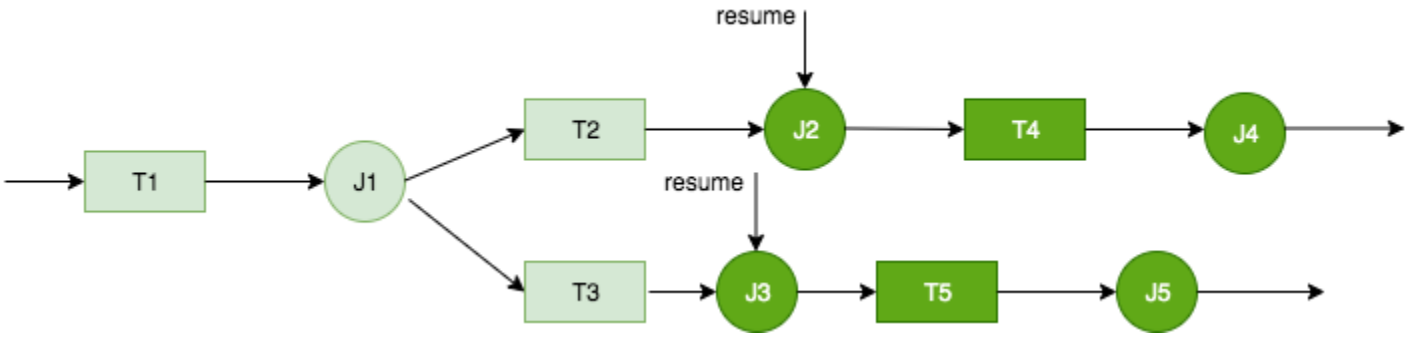
Sehen Sie sich den Workflow W1 im folgenden Diagramm an.



Die Workflow-Ausführung läuft folgendermaßen ab:

1. Auslöser T1 startet Auftrag J1.
2. Der erfolgreiche Abschluss von J1 aktiviert T2 und T3, die die Aufträge J2 bzw. J3 ausführen.
3. J2 und J3 schlagen fehl.
4. Die Auslöser T4 und T5 hängen vom erfolgreichen Abschluss von J2 und J3 ab. Daher werden sie nicht aktiviert und die Aufträge J4 und J5 werden nicht ausgeführt. Workflow W1 wird nur teilweise ausgeführt.

Angenommen, die Probleme, die die Ausführung von J2 und J3 verhindern, werden behoben. J2 und J3 werden als Startpunkte ausgewählt, an denen die Workflow-Ausführung fortgesetzt werden soll.



Die Workflow-Ausführung wird folgendermaßen fortgesetzt:

1. Die Aufträge J2 und J3 werden erfolgreich ausgeführt.
2. Dadurch werden die Auslöser T4 und T5 aktiviert.
3. Die Aufträge J4 und J5 werden erfolgreich ausgeführt.

Die fortgesetzte Workflow-Ausführung wird als separate Ausführung mit einer neuen Ausführungs-ID erfasst. Wenn Sie den Workflow-Verlauf aufrufen, können Sie die vorherige Ausführungs-ID für jede Workflow-Ausführung einsehen. Im Beispiel im folgenden Screenshot hatte die Workflow-Ausführung mit der Ausführungs-ID `wr_c7a22...` (zweite Zeile) einen Knoten, der nicht abgeschlossen wurde. Der Benutzer hat das Problem behoben und die Workflow-Ausführung fortgesetzt, wodurch die Ausführungs-ID `wr_a07e55...` (erste Zeile) generiert wurde.

> **Note**
>
> Für den Rest dieses Artikels bezieht sich der Begriff „fortgesetzte Workflow-Ausführung“ auf die Ausführung, die beim Fortsetzen der vorherigen Workflow-Ausführung erstellt wurde. Die „ursprüngliche Workflow-Ausführung“ bezieht sich auf den Workflow, der nur teilweise ausgeführt wurde und deswegen fortgesetzt werden musste.

Diagramm mit fortgesetzter Workflow-Ausführung

In einer fortgesetzten Workflow-Ausführung ist das Ausführungsdiagramm vollständig, obwohl nur eine Teilmenge von Knoten ausgeführt wird. Die Knoten, die bei der fortgesetzten Workflow-Ausführung nicht ausgeführt wurden, werden daher aus dem Ausführungsdiagramm der ursprünglichen Workflow-Ausführung kopiert. Kopierte Auftrags- und Crawlerknoten, die Teil der ursprünglichen Workflow-Ausführung waren, enthalten Ausführungsdetails.

Betrachten Sie noch einmal den Workflow W1 im vorherigen Diagramm. Wenn die Ausführung mit J2 und J3 fortgesetzt wird, zeigt das Ausführungsdiagramm für die fortgesetzte Workflow-Ausführung alle Aufträge (J1 bis J5) und alle Auslöser (T1 bis T5) an. Die Ausführungsdetails für J1 werden aus der ursprünglichen Workflow-Ausführung kopiert.

Snapshots der Workflow-Ausführung

Wenn ein Workflow-Lauf gestartet wird, AWS Glue erstellt eine Momentaufnahme des Workflow-Entwurfsdiagramms zu diesem Zeitpunkt. Dieser Snapshot wird für die Dauer der Workflow-Ausführung verwendet. Wenn Sie nach dem Start der Ausführung Änderungen an Auslösern

vornehmen, wirken sich diese Änderungen nicht auf die aktuelle Workflow-Ausführung aus. Snapshots stellen sicher, dass Workflow-Ausführungen konsistent ablaufen.

Snapshots machen nur Auslöser unveränderlich. Änderungen, die Sie während der Workflow-Ausführung an nachgelagerten Aufträgen und Crawlern vornehmen, werden für die aktuelle Ausführung wirksam.

Fortsetzen einer Workflow-Ausführung

Führen Sie diese Schritte aus, um eine Workflow-Ausführung fortzusetzen. Sie können eine Workflow-Ausführung fortsetzen, indem Sie den AWS Glue Konsole, API oder AWS Command Line Interface (AWS CLI).

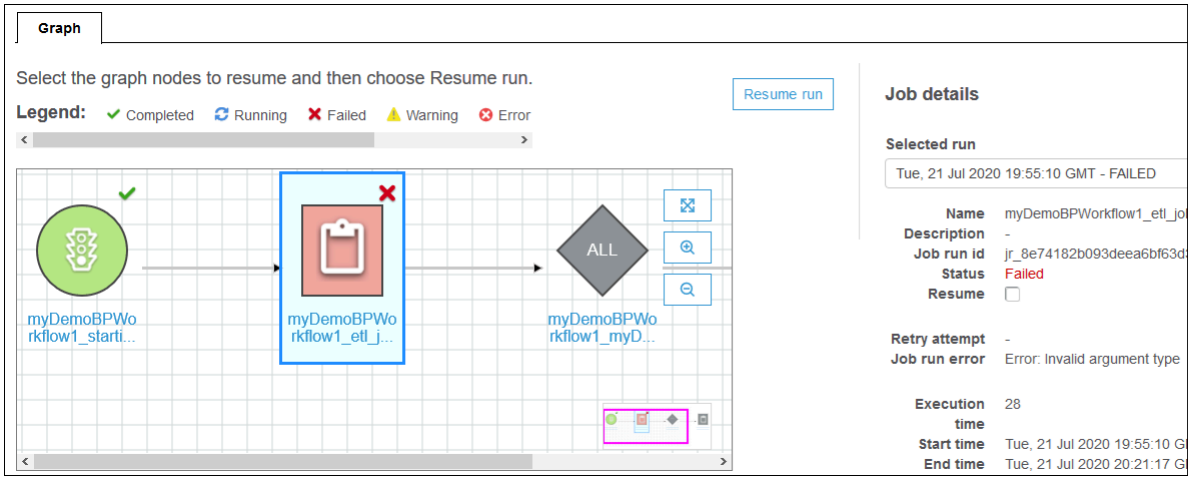
Eine Workflow-Ausführung fortsetzen (Konsole)

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.

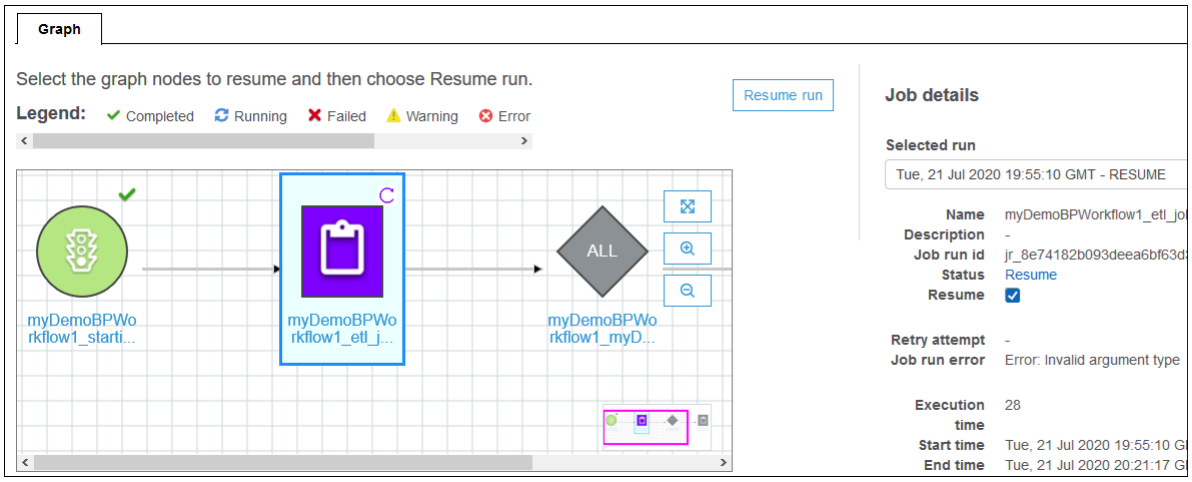
   Melden Sie sich als Benutzer an, der Berechtigungen zum Anzeigen von Workflows und zum Fortsetzung von Workflow-Ausführungen hat.

   > **Note**
   >
   > Um Workflow-Ausführungen wieder aufzunehmen, benötigen Sie die `glue:ResumeWorkflowRun` AWS Identity and Access Management (IAM) - Berechtigung.

2. Wählen Sie im Navigationsbereich die Option Workflows aus.
3. Wählen Sie einen Workflow und dann die Registerkarte History (Verlauf) aus.
4. Wählen Sie einen Workflow aus, der nur teilweise ausgeführt wurde, und gehen Sie auf View run details (Ausführungsdetails anzeigen).
5. Wählen Sie im Ausführungsdiagramm den ersten (oder einzigen) Knoten aus, den Sie neu starten und von dem Sie die Workflow-Ausführung fortsetzen möchten.
6. Wählen Sie im Detailbereich rechts neben dem Diagramm das Kontrollkästchen Resume (Fortsetzen) aus.



Die Farbe des Knotens ändert sich und oben rechts wird ein kleines Fortsetzungssymbol angezeigt.



7. Führen Sie die beiden vorherigen Schritte aus, um weitere Knoten neu zu starten.
8. Wählen Sie erneut Resume (Fortsetzen) aus.

Eine Workflow-Ausführung fortsetzen (AWS CLI)

1. Stellen Sie sicher, dass Sie die IAM-Berechtigung `glue:ResumeWorkflowRun` haben.
2. Rufen Sie den Knoten IDs für die Knoten ab, die Sie neu starten möchten.

   a. Führen Sie für die ursprüngliche Workflow-Ausführung den Befehl `get-workflow-run` aus. Geben Sie den Workflow-Namen und die Ausführungs-ID an und fügen Sie die Option `--include-graph` hinzu, wie im folgenden Beispiel veranschaulicht. Rufen Sie die Ausführungs-ID von der Registerkarte History (Verlauf) in der Konsole ab oder führen Sie dazu den Befehl `get-workflow` aus.

```
aws glue get-workflow-run --name cloudtrailtest1 --run-id
 wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --include-
graph
```

Der Befehl gibt die Knoten und Edges des Diagramms als großes JSON-Objekt zurück.

b. Suchen Sie die relevanten Knoten anhand der `Type`- und `Name`-Eigenschaften der Knotenobjekte.

Das folgende Beispiel zeigt ein Knotenobjekt aus der Ausgabe.

```
{
    "Type": "JOB",
    "Name": "test1_post_failure_4592978",
    "UniqueId":
 "wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd",
    "JobDetails": {
        "JobRuns": [
            {
                "Id":
 "jr_690b9f7fc5cb399204bc542c6c956f39934496a5d665a42de891e5b01f59e613",
                "Attempt": 0,
                "TriggerName": "test1_aggregate_failure_649b2432",
                "JobName": "test1_post_failure_4592978",
                "StartedOn": 1595358275.375,
                "LastModifiedOn": 1595358298.785,
                "CompletedOn": 1595358298.785,
                "JobRunState": "FAILED",
                "PredecessorRuns": [],
                "AllocatedCapacity": 0,
                "ExecutionTime": 16,
                "Timeout": 2880,
                "MaxCapacity": 0.0625,
                "LogGroupName": "/aws-glue/python-jobs"
            }
        ]
    }
}
```

c. Rufen Sie die Knoten-ID aus der `UniqueId`-Eigenschaft des Knotenobjekts ab.

3. Führen Sie den Befehl `resume-workflow-run` aus. Geben Sie den Workflow-Namen, die Ausführungs-ID und die durch Leerzeichen IDs getrennte Liste der Knoten an, wie im folgenden Beispiel gezeigt.

```
aws glue resume-workflow-run --name cloudtrailtest1 --run-id
 wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --node-
ids wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3
 wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd
```

Der Befehl gibt die Ausführungs-ID der fortgesetzten (neuen) Workflow-Ausführung und eine Liste der Knoten aus, die gestartet werden.

```
{
    "RunId": "wr_2ada0d3209a262fc1156e4291134b3bd643491bcfb0ceead30bd3e4efac24de9",
    "NodeIds": [
        "wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3"
    ]
}
```

Beachten Sie Folgendes: Obwohl der Beispielbefehl `resume-workflow-run` zwei Knoten aufgeführt hat, die neu gestartet werden sollen, zeigt die Beispielausgabe an, dass nur ein Knoten neu gestartet werden würde. Der Grund dafür ist, dass ein Knoten dem anderen Knoten nachgelagert war und der nachgelagerte Knoten durch den normalen Ablauf des Workflows trotzdem neu gestartet würde.

Hinweise und Einschränkungen für die Fortsetzung von Workflow-Ausführungen

Beachten Sie die folgenden Hinweise und Einschränkungen, wenn Sie Workflow-Ausführungen fortsetzen.

- Sie können eine Workflow-Ausführung nur fortsetzen, wenn sie sich im Zustand `COMPLETED` befindet.

> **Note**
>
> Selbst wenn ein oder mehrere Knoten in einer Workflow-Ausführung nicht abgeschlossen werden, wird der Workflow-Ausführungsstatus als `COMPLETED` angezeigt. Sie sollten

das Ausführungsdiagramm überprüfen, um alle Knoten zu ermitteln, die nicht erfolgreich abgeschlossen wurden.

- Sie können eine Workflow-Ausführung von jedem Auftrags- oder Crawlerknoten fortsetzen, der bei der ursprünglichen Workflow-Ausführung ausgeführt werden sollte. Ein Workflow, der von einem Auslöserknoten ausgeführt wird, kann jedoch nicht fortgesetzt werden.
- Beim Neustart eines Knotens wird sein Zustand nicht zurückgesetzt. Alle Daten, die teilweise verarbeitet wurden, werden nicht zurückgesetzt.
- Sie können dieselbe Workflow-Ausführung mehrmals fortsetzen. Wenn eine fortgesetzte Workflow-Ausführung nur teilweise ausgeführt wird, können Sie das Problem beheben und die fortgesetzte Ausführung fortsetzen.
- Wenn Sie zwei Knoten zum Neustart auswählen und sie voneinander abhängig sind, wird der vorgelagerte Knoten vor dem nachgelagerten Knoten ausgeführt. Im Grunde genommen ist die Auswahl des nachgelagerten Knotens redundant, da er gemäß dem normalen Ablauf des Workflows ausgeführt wird.

Eigenschaften für Workflow-Ausführungen abrufen und festlegen in AWS Glue

Verwenden Sie die Eigenschaften der Workflow-Ausführung, um den Status der Jobs in Ihrem AWS Glue Arbeitsablauf. Sie können Standard-Ausführungseigenschaften festlegen, wenn Sie den Workflow erstellen. Wenn die Aufträge ausgeführt werden, können sie die Werte der Ausführungseigenschaft abrufen und optional ändern, damit sie als Eingabe für Aufträge später im Workflow verwendet werden können. Wenn ein Auftrag eine Ausführungseigenschaft ändert, ist der neue Wert nur für die Ausführung des Workflows vorhanden. Die Standard-Ausführungseigenschaften sind nicht betroffen.

Wenn Ihr AWS Glue-Job nicht Teil eines Workflows ist, werden diese Eigenschaften nicht festgelegt.

Der folgende Python-Beispielcode aus einem ETL-Auftrag (Extrahieren, Transformieren und Laden) zeigt, wie die Ausführungseigenschaften für den Workflow abgerufen werden.

```
import sys
import boto3
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from awsglue.context import GlueContext
```

```
from pyspark.context import SparkContext

glue_client = boto3.client("glue")
args = getResolvedOptions(sys.argv, ['JOB_NAME','WORKFLOW_NAME', 'WORKFLOW_RUN_ID'])
workflow_name = args['WORKFLOW_NAME']
workflow_run_id = args['WORKFLOW_RUN_ID']
workflow_params = glue_client.get_workflow_run_properties(Name=workflow_name,
                                       RunId=workflow_run_id)["RunProperties"]

target_database = workflow_params['target_database']
target_s3_location = workflow_params['target_s3_location']
```

Im folgenden Code wird die Ausführungseigenschaft `target_format` dann auf `'csv'` festgelegt.

```
workflow_params['target_format'] = 'csv'
glue_client.put_workflow_run_properties(Name=workflow_name, RunId=workflow_run_id,
 RunProperties=workflow_params)
```

Weitere Informationen finden Sie hier:

- GetWorkflowRunProperties Aktion (Python: get_workflow_run_properties)
- PutWorkflowRunProperties Aktion (Python: put_workflow_run_properties)

Abfragen von Workflows mit dem AWS Glue API

AWS Glue bietet eine umfangreiche API für die Verwaltung von Workflows. Sie können mit der AWS Glue API eine statische Ansicht eines Workflows oder eine dynamische Ansicht eines aktiven Workflows abrufen. Weitere Informationen finden Sie unter Workflows.

Themen

- Abfragen von statischen Ansichten
- Abfragen von dynamischen Ansichten

Abfragen von statischen Ansichten

Verwenden Sie die API-Operation `GetWorkflow`, um eine statische Ansicht mit der Struktur des Workflows abzurufen. Diese Operation gibt ein Diagramm aus Knoten und Edges zurück, wobei jeder Knoten für einen Auslöser, einen Auftrag oder einen Crawler steht. Edges definieren die Beziehungen zwischen Knoten. Sie werden durch Konnektoren (Pfeile) in der Grafik in der AWS Glue console.

Sie können diesen Vorgang auch mit gängigen Bibliotheken zur Grafikverarbeitung wie NetworkX, igraph, JGraph T und dem Java Universal Network/Graph (JUNG) Framework verwenden. Da alle diese Bibliotheken Diagramme ähnlich darstellen, werden nur minimale Transformationen benötigt.

Die von dieser API zurückgegebene statische Ansicht entspricht der neuesten Definition der mit dem up-to-date Workflow verknüpften Trigger am häufigsten.

Diagrammdefinition

Bei einem Workflow-Diagramm (G) handelt es sich um ein geordnetes Paar (N, E), wobei N eine Gruppe von Knoten darstellt und E eine Reihe von Edges. Ein Knoten ist ein Eckpunkt im Diagramm, der mit einer eindeutigen Nummer gekennzeichnet ist. Typen von Knoten sind Auslöser, Aufträge oder Crawler. Beispiel: `{name:T1, type:Trigger, uniqueId:1}, {name:J1, type:Job, uniqueId:2}`.

Ein Edge ist ein 2-Tupel in der Form (`src, dest`), wobei `src` und `dest` Knoten sind und es eine gerichtete Verbindung von `src` zu `dest` gibt.

Beispiel zum Abfragen einer statischen Ansicht

Betrachten Sie einen bedingten Auslöser T, der den Auftrag J2 nach Abschluss des Auftrags J1 auslöst.

```
J1 ---> T ---> J2
```

Knoten: J1, T, J2

Edges: (J1, T), (T, J2)

Abfragen von dynamischen Ansichten

Mit der API-Operation `GetWorkflowRun` können Sie eine dynamische Ansicht für einen aktiven Workflow abrufen. Diese Operation gibt dieselbe statische Ansicht des Diagramms sowie Metadaten im Zusammenhang mit der Workflow-Ausführung zurück.

Knoten, die im Aufruf `GetWorkflowRun` Aufträge darstellen, verfügen über eine Liste der Auftragsausführungen, die als Teil der letzten Ausführung des Workflows initiiert wurden. Mit dieser Liste können Sie den Ausführungsstatus für jeden Auftrag im Diagramm anzeigen. Für nachgelagerte Abhängigkeiten, die noch nicht ausgeführt wurden, ist dieses Feld auf `null` festgelegt. Anhand der im Diagramm dargestellten Informationen sehen Sie den aktuellen Status jedes Workflows zu jedem beliebigen Zeitpunkt.

Die dynamische Ansicht, die von dieser API zurückgegeben wird, basiert auf der statischen Ansicht, die beim Starten der Workflow-Ausführung vorhanden war.

Beispiel zu Laufzeitknoten: `{name:T1, type: Trigger, uniqueId:1}`, `{name:J1, type:Job, uniqueId:2, jobDetails:{jobRuns}}`, `{name:C1, type:Crawler, uniqueId:3, crawlerDetails:{crawls}}`

Beispiel 1: Dynamische Ansicht

Das folgende Beispiel zeigt einen einfachen Workflow mit zwei Auslösern.

- Knoten: t1, j1, t2, j2
- Edges: (t1, j1), (j1, t2), (t2, j2)

Die `GetWorkflow`-Antwort enthält Folgendes.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1
        },
        {
            "type" : Job,
            "name" : "j1",
            "uniqueId" : 2
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3
        },
        {
            "type" : Job,
            "name" : "j2",
            "uniqueId" : 4
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
```

```
            "destinationId" : 2
        },
        {
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
}
```

Die `GetWorkflowRun`-Antwort enthält Folgendes.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
        {
            "type" : Job,
            "name" : "j1",
            "uniqueId" : 2,
            "jobDetails" : [
                {
                    "id" : "jr_12334",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
```

```
        {
            "type" : Job,
            "name" : "j2",
            "uniqueId" : 4,
            "jobDetails" : [
                {
                    "id" : "jr_1233sdf4",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
            "destinationId" : 2
        },
        {
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
}
```

Beispiel 2: Mehrere Aufträge mit einem bedingten Auslöser

Das folgende Beispiel zeigt einen Workflow mit mehreren Aufträgen und einem bedingten Auslöser (t3).

```
Consider Flow:
T(t1) ---> J(j1) ---> T(t2) ---> J(j2)
             |                    |
             |                    |
             >+------> T(t3) <-----+
                        |
                        |
                      J(j3)
```

```
Graph generated:
Nodes: t1, t2, t3, j1, j2, j3
Edges: (t1, j1), (j1, t2), (t2, j2), (j1, t3), (j2, t3), (t3, j3)
```

Blueprint- und Workflow-Einschränkungen in AWS Glue

Folgende Einschränkungen gelten für Blueprints und Workflows.

Einschränkungen für Blueprints

Berücksichtigen Sie die folgenden Einschränkungen für Blueprints:

- Der Blueprint muss in derselben AWS Region registriert sein, in der sich der Amazon S3 S3-Bucket befindet.
- Um Blueprints für mehrere AWS Konten gemeinsam zu nutzen, müssen Sie die Leseberechtigungen für das Blueprint-ZIP-Archiv in Amazon S3 erteilen. Kunden, die über Leseberechtigungen für ein Blueprint-ZIP-Archiv verfügen, können den Blueprint in ihrem AWS Konto registrieren und verwenden.
- Die Blueprint-Parameter werden als einzelnes JSON-Objekt gespeichert. Die maximale Länge dieses Objekts beträgt 128 KB.
- Die maximale unkomprimierte Größe des Blueprint-ZIP-Archivs beträgt 5 MB. Die maximale Größe beträgt 1 MB.
- Beschränken Sie die Gesamtzahl der Aufträge, Crawler und Trigger innerhalb eines Workflows auf 100 oder weniger. Wenn Sie mehr als 100 einschließen, werden möglicherweise Fehler angezeigt, wenn Sie versuchen, Workflow-Läufe fortzusetzen oder zu beenden.

Einschränkungen für Workflows

Berücksichtigen Sie die folgenden Einschränkungen für Workflows. Einige dieser Kommentare richten sich eher an einen Benutzer, der Workflows manuell erstellt.

- Die maximale Batchgröße für einen EventBridge Amazon-Event-Trigger beträgt 100. Die maximale Fenstergröße beträgt 900 Sekunden (15 Minuten).
- Ein Auslöser kann nur einem Workflow zugeordnet werden.
- Es ist nur ein Startauslöser (On-Demand- oder Zeitplanauslöser) zulässig.

- Wenn ein Auftrag oder Crawler in einem Workflow von einem Auslöser gestartet wird, der sich außerhalb des Workflows befindet, lösen Auslöser innerhalb des Workflows, die vom Abschluss des Auftrags oder des Crawlers abhängen (erfolgreich oder anderweitig), nicht aus.
- Gleiches gilt, wenn ein Auftrag oder Crawler in einem Workflow über Auslöser verfügt, die vom Abschluss des Auftrags oder Crawlers (erfolgreich oder nicht erfolgreich) sowohl innerhalb als auch außerhalb des Workflows abhängen. Wenn der Auftrag oder Crawler von innerhalb eines Workflows gestartet wird, werden bei Abschluss des Auftrags oder Crawlers nur die Auslöser innerhalb des Workflows aktiviert.

Behebung von Blueprint-Fehlern in AWS Glue

Wenn Sie bei der Verwendung auf Fehler stoßen AWS Glue Verwenden Sie die folgenden Lösungen, um die Ursache der Probleme zu finden und sie zu beheben.

Themen

- Fehler: Modul fehlt PySpark
- Fehler: Blueprint-Konfigurationsdatei fehlt
- Fehler: fehlende importierte Datei
- Fehler iamPassRole : nicht autorisiert, auf der Ressource zu arbeiten
- Fehler: Ungültiger Cron-Zeitplan
- Fehler: Ein Auslöser mit demselben Namen ist bereits vorhanden
- Fehler: Workflow mit Name: foo ist bereits vorhanden.
- Fehler: Modul im angegebenen layoutGenerator-Pfad nicht gefunden
- Fehler: Validierungsfehler im Feld „Connections“ (Verbindungen)

Fehler: Modul fehlt PySpark

AWS Glue gibt den Fehler „Unbekannter Fehler beim Ausführen der Layoutgeneratorfunktion ModuleNotFoundError: Kein Modul namens 'pyspark'“ zurück.

Wenn Sie das Blueprint-Archiv entpacken, sind zwei Szenarien möglich:

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg
```

```
  inflating: compaction/layout.py
  inflating: compaction/README.md
  inflating: compaction/compaction.py

$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
  inflating: compaction.py
  inflating: layout.py
  inflating: README.md
```

Im ersten Fall wurden alle zugehörigen Blueprint-Dateien unter einem Ordner namens „compaction“ (Komprimierung) abgelegt, der dann in eine ZIP-Datei namens compaction.zip konvertiert wurde.

Im zweiten Fall wurden alle für den Blueprint benötigten Dateien nicht in einen Ordner aufgenommen und als Stammdateien in die ZIP-Datei compaction.zip eingefügt.

Das Erstellen einer Datei in einem der oben genannten Formate ist erlaubt. Stellen Sie jedoch sicher, dass `blueprint.cfg` den richtigen Pfad zum Namen der Funktion in dem Skript hat, das das Layout generiert.

Beispiele

In Fall 1: `blueprint.cfg` sollte `layoutGenerator` wie folgt enthalten:

```
layoutGenerator": "compaction.layout.generate_layout"
```

In Fall 2: `blueprint.cfg` sollte `layoutGenerator` wie folgt enthalten:

```
layoutGenerator": "layout.generate_layout"
```

Wenn dieser Pfad nicht korrekt enthalten ist, könnte wie angegeben ein Fehler angezeigt werden. Wenn Sie beispielsweise die Ordnerstruktur wie in Fall 2 haben und den `layoutGenerator` wie in Fall 1, sehen Sie wahrscheinlich die obige Fehlermeldung.

Fehler: Blueprint-Konfigurationsdatei fehlt

AWS Glue gibt den Fehler „Unbekannter Fehler beim Ausführen der Layoutgenerator-Funktion FileNotFoundError: [Errno 2] Keine solche Datei oder kein solches Verzeichnis: '/ tmp/compaction/ blueprint .cfg'“ zurück.

Die blueprint.cfg sollte auf der Stammebene des ZIP-Archivs oder in einem Ordner platziert werden, der den gleichen Namen wie das ZIP-Archiv hat.

Wenn wir das Blueprint-ZIP-Archiv extrahieren, wird erwartet, dass sich die blueprint.cfg in einem der folgenden Pfade befindet. Wenn es nicht in einem der folgenden Pfade gefunden wird, sehen Sie wahrscheinlich die obige Fehlermeldung.

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg

$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
```

Fehler: fehlende importierte Datei

AWS Glue gibt den Fehler „Unbekannter Fehler beim Ausführen der Layoutgenerator-Funktion FileNotFoundError: [Errno 2] Keine solche Datei oder kein solches Verzeichnis: * *'demo-project/foo.py'“ zurück.

Wenn Ihr Skript zur Layoutgenerierung Funktionen zum Lesen anderer Dateien hat, stellen Sie sicher, dass Sie einen vollständigen Pfad für die zu importierende Datei angeben. Beispielsweise kann das Skript „Conversion.py“ in Layout.py referenziert werden. Weitere Informationen finden Sie unter Blueprint-Beispielprojekt.

Fehler iamPassRole : nicht autorisiert, auf der Ressource zu arbeiten

AWS Glue gibt den Fehler „User: arn:aws:sts: :123456789012:assumed-“ zurück role/AWSGlueServiceRole/GlueSession is not authorized to perform: iam:PassRole on resource: arn:aws:iam::123456789012:role/AWSGlueServiceRole

Wenn die Aufträge und Crawler im Workflow dieselbe Rolle übernehmen, die zum Erstellen des Workflows vom Blueprint übergeben wurde, muss die Blueprint-Rolle die Berechtigung `iam:PassRole` für sich selbst enthalten.

Wenn die Aufträge und Crawler im Workflow eine Rolle übernehmen, die sich von der Rolle unterscheidet, die zum Erstellen der Entitäten des Workflows vom Blueprint übergeben wurde, muss die Blueprint-Rolle die Berechtigung `iam:PassRole` für diese andere Rolle enthalten.

Weitere Informationen finden Sie unter Berechtigungen für Blueprint-Rollen.

Fehler: Ungültiger Cron-Zeitplan

AWS Glue gibt den Fehler „Der Zeitplan cron (0 0 * * * *) ist ungültig“ zurück.

Stellen Sie einen gültigen cron-Ausdruck bereit. Weitere Informationen finden Sie unter Zeitbasierte Pläne für Aufträge und Crawler.

Fehler: Ein Auslöser mit demselben Namen ist bereits vorhanden

AWS Glue gibt den Fehler „Trigger mit dem Namen 'foo_starting_trigger' wurde bereits mit einer anderen Konfiguration gesendet“ zurück.

Für einen Blueprint müssen Sie keine Auslöser im Layoutskript für die Workflow-Erstellung definieren. Die Auslösererstellung wird von der Blueprint-Bibliothek verwaltet und basiert auf den Abhängigkeiten, die zwischen zwei Aktionen definiert sind.

Die Auslöser werden wie folgt benannt:

- Für den Startauslöser im Workflow lautet das Namensformat <workflow_name>_starting_trigger.
- Für einen Knoten (Job/Crawler) im Workflow, der von der Fertigstellung eines oder mehrerer Upstream-Knoten abhängt; AWS Glue <workflow_name><node_name>definiert einen Trigger mit dem Namen _ _trigger

Dieser Fehler bedeutet, dass bereits ein Auslöser mit demselben Namen vorhanden ist. Sie können den vorhandenen Auslöser löschen und die Workflow-Erstellung erneut ausführen.

> **Note**
>
> Beim Löschen eines Workflows werden die Knoten im Workflow nicht gelöscht. Es ist möglich, dass beim Löschen eines Workflows Auslöser zurückbleiben. Sie können deshalb nicht die Fehlermeldung erhalten, dass ein Workflow bereits vorhanden ist. Es ist aber möglich, dass Sie die Fehlermeldung erhalten, dass ein Auslöser bereits vorhanden ist. Dies kann passieren, wenn Sie einen Workflow erstellen, wieder löschen und dann versuchen, ihn mit demselben Namen auf der Grundlage desselben Blueprints neu zu erstellen.

Fehler: Workflow mit Name: foo ist bereits vorhanden.

Der Workflow-Name muss eindeutig sein. Geben Sie einen anderen Namen ein.

Fehler: Modul im angegebenen layoutGenerator-Pfad nicht gefunden

AWS Glue gibt den Fehler „Unbekannter Fehler beim Ausführen der Layoutgenerator-Funktion ModuleNotFoundError: Kein Modul namens 'crawl_s3_locations'“ zurück.

```
layoutGenerator": "crawl_s3_locations.layout.generate_layout"
```

Wenn Sie beispielsweise den obigen layoutGenerator-Pfad haben, muss dieser beim Entpacken des Blueprint-Archivs wie folgt aussehen:

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
   creating: crawl_s3_locations/
  inflating: crawl_s3_locations/blueprint.cfg
  inflating: crawl_s3_locations/layout.py
  inflating: crawl_s3_locations/README.md
```

Wenn Sie ein Blueprint-Archiv entpacken, das wie folgt aussieht, können Sie den obigen Fehler erhalten.

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
  inflating: blueprint.cfg
  inflating: layout.py
  inflating: README.md
```

Sie können sehen, dass kein Ordner namens `crawl_s3_locations` vorhanden ist, und wenn der Pfad `layoutGenerator` über das Modul`crawl_s3_locations` auf die Layoutdatei verweist, können Sie die obige Fehlermeldung erhalten.

Fehler: Validierungsfehler im Feld „Connections“ (Verbindungen)

AWS Glue <class 'dict'>gibt den Fehler „Unbekannter Fehler bei der Ausführung der Layoutgenerator-Funktion zurück TypeError: Der Wert ['foo'] für die Tastenverbindungen sollte vom Typ sein!“.

Dies ist ein Validierungsfehler. Im Feld `Connections` in der `Job`-Klasse wird statt dem erwarteten Wörterbuch eine Liste von Werten bereitgestellt. Dies verursacht den Fehler.

```
User input was list of values
Connections= ['string']

Should be a dict like the following
Connections*=*{'Connections': ['string']}
```

Um diese Laufzeitfehler beim Erstellen eines Workflows anhand eines Blueprints zu vermeiden, können Sie die Workflow-, Auftrags- und Crawlerdefinitionen überprüfen, wie in Testen eines Blueprints beschrieben.

Die Syntax finden Sie unter AWS Glue Blueprint-Klassenreferenz zur Definition von AWS Glue Job, Crawler und Workflow im Layout-Skript.

Berechtigungen für Personas und Rollen für AWS Glue-Blueprints

Im Folgenden sind die typischen Personas und die empfohlenen AWS Identity and Access Management (IAM-) Berechtigungsrichtlinien für Personas und Rollen für Blueprints aufgeführt. AWS Glue

Themen

- Blueprint-Personas
- Berechtigungen für Blueprint-Personas
- Berechtigungen für Blueprint-Rollen

Blueprint-Personas

Im Folgenden werden die Personas aufgeführt, die normalerweise am Lebenszyklus von AWS Glue-Blueprints beteiligt sind.

Persona	Beschreibung
AWS GlueDeveloper	Entwickelt, testet und veröffentlicht Blueprints.
AWS Glue-Administrator	Registriert, verwaltet und erteilt Berechtigungen für Blueprints.

Persona	Beschreibung
Datenanalyst	Führt Blueprints aus, um Workflows zu erstellen.

Weitere Informationen finden Sie unter the section called “Übersicht über Blueprints”.

Berechtigungen für Blueprint-Personas

Im Folgenden finden Sie die empfohlenen Berechtigungen für die jeweiligen Blueprint-Personas.

AWS Glue-Entwicklerberechtigungen für Blueprints

Der AWS Glue-Entwickler muss Schreibberechtigungen für den Amazon-S3-Bucket haben, der zum Veröffentlichen des Blueprints verwendet wird. Oft registriert der Entwickler den Blueprint, nachdem er ihn hochgeladen hat. In diesem Fall benötigt der Entwickler die Berechtigungen, die in the section called “AWS Glue-Administratorberechtigungen für Blueprints” aufgeführt werden. Wenn der Entwickler den Blueprint nach der Registrierung testen möchte, benötigt er außerdem die Berechtigungen unter the section called “Datenanalystenberechtigungen für Blueprints”.

AWS Glue-Administratorberechtigungen für Blueprints

Mit der folgenden Richtlinie werden Berechtigungen zum Registrieren, Anzeigen und Verwalten von AWS Glue-Blueprints erteilt.

> **Important**
>
> Ersetzen Sie in der folgenden Richtlinie *<s3-bucket-name>* und *<prefix>* durch den Amazon S3 S3-Pfad zu den hochgeladenen Blueprint-ZIP-Archiven, um sich zu registrieren.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "glue:CreateBlueprint",
                "glue:UpdateBlueprint",
                "glue:DeleteBlueprint",
                "glue:GetBlueprint",
                "glue:ListBlueprints",
                "glue:BatchGetBlueprints"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::<s3-bucket-name>/<prefix>/*"
        }
    ]
}
```

Datenanalystenberechtigungen für Blueprints

Die folgende Richtlinie erteilt Berechtigungen zum Ausführen von Blueprints und zum Anzeigen des resultierenden Workflow- bzw. der Workflow-Komponenten. Außerdem gewährt sie `PassRole` für die Rolle, die AWS Glue übernimmt, um den Workflow und die Workflow-Komponenten zu erstellen.

Die Richtlinie gewährt Berechtigungen für jede Ressource. Wenn Sie einen differenzierten Zugriff auf einzelne Blueprints konfigurieren möchten, verwenden Sie das folgende Format für Blueprints: ARNs

```
arn:aws:glue:<region>:<account-id>:blueprint/<blueprint-name>
```

> **Important**
>
> Ersetzen Sie in der folgenden Richtlinie durch ein gültiges AWS Konto und *<account-id>* *<role-name>* ersetzen Sie es durch den Namen der Rolle, die zum Ausführen eines Blueprints verwendet wird. Die Berechtigungen, die für diese Rolle erforderlich sind, finden Sie unter the section called “Berechtigungen für Blueprint-Rollen”.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListBlueprints",
                "glue:GetBlueprint",
                "glue:StartBlueprintRun",
                "glue:GetBlueprintRun",
                "glue:GetBlueprintRuns",
                "glue:GetCrawler",
                "glue:ListTriggers",
                "glue:ListJobs",
                "glue:BatchGetCrawlers",
                "glue:GetTrigger",
                "glue:BatchGetWorkflows",
                "glue:BatchGetTriggers",
                "glue:BatchGetJobs",
                "glue:BatchGetBlueprints",
                "glue:GetWorkflowRun",
                "glue:GetWorkflowRuns",
                "glue:ListCrawlers",
                "glue:ListWorkflows",
                "glue:GetJob",
                "glue:GetWorkflow",
                "glue:StartWorkflowRun"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<account-id>:role/<role-name>"
        }
    ]
}
```

Berechtigungen für Blueprint-Rollen

Im Folgenden sehen Sie die vorgeschlagenen Berechtigungen für die IAM-Rolle, die zum Erstellen eines Workflows aus einem Blueprint verwendet wird. Die Rolle muss über eine Vertrauensstellung bei `glue.amazonaws.com` verfügen.

> **Important**
>
> Ersetzen Sie es in der folgenden Richtlinie *<account-id>* durch ein gültiges AWS Konto und *<role-name>* ersetzen Sie es durch den Namen der Rolle.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateJob",
                "glue:GetCrawler",
                "glue:GetTrigger",
                "glue:DeleteCrawler",
                "glue:CreateTrigger",
                "glue:DeleteTrigger",
                "glue:DeleteJob",
                "glue:CreateWorkflow",
                "glue:DeleteWorkflow",
                "glue:GetJob",
                "glue:GetWorkflow",
                "glue:CreateCrawler"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<account-id>:role/<role-name>"
        }
    ]
```

```
}
```

> **Note**
>
> Wenn die Aufträge und Crawler im Workflow eine andere als diese Rolle übernehmen, muss diese Richtlinie die `iam:PassRole`-Berechtigung für die entsprechende Rolle einbinden, statt für die Blueprint-Rolle.

Entwicklung von Bauplänen in AWS Glue

Ihre Organisation verfügt möglicherweise über eine Reihe ähnlicher ETL-Anwendungsfälle, die davon profitieren können, dass sie einen einzelnen Workflow parametrisieren können, um sie alle zu verarbeiten. Um diesem Bedarf gerecht zu werden, AWS Glue ermöglicht es Ihnen, Blueprints zu definieren, die Sie zur Generierung von Workflows verwenden können. Ein Blueprint akzeptiert Parameter. Ein Datenanalyst kann aus einem einzelnen Blueprint verschiedene Workflows ableiten, um ähnliche ETL-Anwendungsfälle zu verarbeiten. Einen fertigen Blueprint können Sie für verschiedene Abteilungen, Teams und Projekte wiederverwenden.

Themen

- Überblick über Baupläne in AWS Glue
- Entwicklung von Bauplänen in AWS Glue
- Registrierung eines Blueprints in AWS Glue
- Baupläne anzeigen in AWS Glue
- Aktualisierung eines Blueprints in AWS Glue
- Erstellen eines Workflows aus einem Blueprint in AWS Glue
- Das Anzeigen von Blueprint läuft in AWS Glue

Überblick über Baupläne in AWS Glue

> **Note**
>
> Die Blueprints-Funktion ist derzeit in den folgenden Regionen der AWS Glue-Konsole nicht verfügbar: Asien-Pazifik (Jakarta) und Naher Osten (VAE).

AWS Glue Blueprints bieten eine Möglichkeit, sie zu erstellen und zu teilen AWS Glue Arbeitsabläufe. Wenn es einen komplexen ETL-Prozess gibt, der für ähnliche Anwendungsfälle verwendet werden könnte, anstatt einen AWS Glue Für jeden Anwendungsfall können Sie einen einzelnen Blueprint erstellen.

Im Blueprint sind die Aufträge und Crawler für den Workflow sowie die Parameter definiert, die der Workflow-Benutzer beim Ausführen des Blueprints angibt, um einen Workflow zu erstellen. Durch die Verwendung von Parametern lassen sich aus einem einzelnen Blueprint Workflows für verschiedene ähnliche Anwendungsfälle generieren. Weitere Informationen zu Workflows finden Sie unter Überblick über Workflows in AWS Glue.

Im Folgenden sind Sie einige Anwendungsfälle für Blueprints beschrieben:

- Sie möchten einen vorhandenen Datensatz partitionieren. Die Eingabeparameter für den Blueprint sind Amazon-Simple-Storage-Service(Amazon S3)-Quell- und -Zielpfade sowie eine Liste von Partitionsspalten.
- Sie möchten einen Snapshot einer Amazon-DynamoDB-Tabelle in einem SQL-Datenspeicher wie Amazon Redshift erstellen. Die Eingabeparameter für den Blueprint sind der DynamoDB-Tabellenname und ein AWS Glue Verbindung, die einen Amazon Redshift Redshift-Cluster und eine Zieldatenbank bezeichnet.
- Sie möchten CSV-Daten in mehreren Amazon-S3-Pfaden in Parquet konvertieren. Du willst AWS Glue Der Workflow muss für jeden Pfad einen separaten Crawler und einen separaten Job enthalten. Die Eingabeparameter sind die Zieldatenbank in AWS Glue Datenkatalog und eine kommagetrennte Liste von Amazon S3 S3-Pfaden. Beachten Sie, dass in diesem Fall die Anzahl der Crawler und Aufträge, die der Workflow erstellt, variabel ist.

Komponenten von Blueprints

Ein Blueprint ist ein ZIP-Archiv, das die folgenden Komponenten enthält:

- Ein Python-Layoutgenerator-Skript

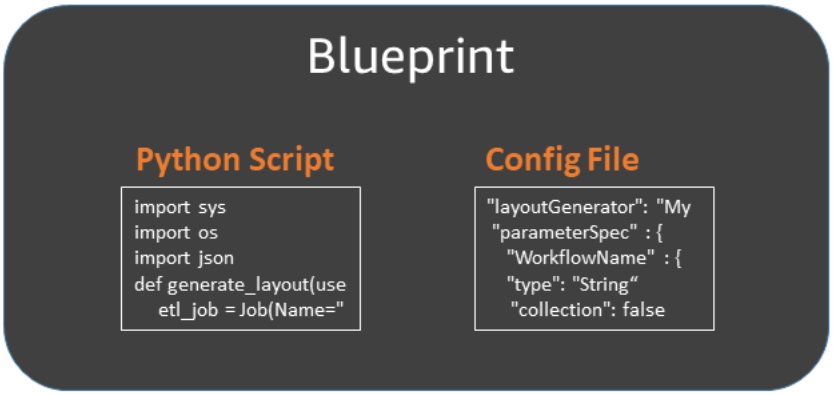
  Enthält eine Funktion, die das Workflow-Layout angibt – die Crawler und Aufträge, die für den Workflow erstellt werden sollen, die Auftrags- und Crawlereigenschaften sowie die Abhängigkeiten zwischen den Aufträgen und Crawlern. Die Funktion akzeptiert Blueprint-Parameter und gibt eine Workflow-Struktur (JSON-Objekt) zurück, die AWS Glue verwendet, um den Workflow zu generieren. Da Sie ein Python-Skript zum Generieren des Workflows verwenden, können Sie Ihre eigene Logik hinzufügen, die auf Ihre Anwendungsfälle abgestimmt ist.

- Eine Konfigurationsdatei

  Gibt den vollständig qualifizierten Namen der Python-Funktion an, die das Workflow-Layout generiert. Gibt außerdem die Namen, Datentypen und andere Eigenschaften aller Blueprint-Parameter an, die vom Skript verwendet werden.

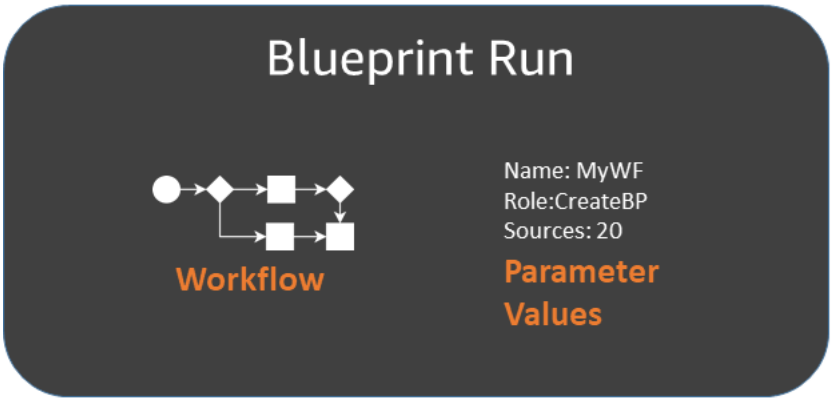
- (Optional) ETL-Skripte und unterstützende Dateien

  In einem erweiterten Anwendungsfall können Sie den Speicherort der ETL-Skripte, die Ihre Aufträge verwenden, parametrisieren. Sie können Auftragsskriptdateien in das ZIP-Archiv aufnehmen und einen Blueprint-Parameter für einen Amazon-S3-Speicherort angeben, in den die Skripte kopiert werden sollen. Das Layout-Generator-Skript kann die ETL-Skripte an den angegebenen Speicherort kopieren und diesen Speicherort als Pfadeigenschaft für das Auftragsskript angeben. Sie können auch Bibliotheken oder andere unterstützende Dateien einschließen, vorausgesetzt, Ihr Skript kann sie verarbeiten.



Blueprint-Ausführungen

Wenn Sie einen Workflow anhand eines Blueprints erstellen, AWS Glue führt den Blueprint aus, der einen asynchronen Prozess startet, um den Workflow und die Jobs, Crawler und Trigger zu erstellen, die der Workflow kapselt. AWS Glue verwendet den Blueprint-Lauf, um die Erstellung des Workflows und seiner Komponenten zu orchestrieren. Der Ausführungsstatus des Blueprints informiert Sie über den Status des Erstellungsprozesses. Die Blueprint-Ausführung speichert auch die Werte, die Sie als Blueprint-Parameter angegeben haben.



Sie können Blueprint-Läufe mit dem AWS Glue Konsole oder AWS Command Line Interface (AWS CLI). Bei der Problembehandlung eines Workflows können Sie jederzeit die Blueprint-Ausführung anzeigen, um die Parameterwerte einzusehen, die zum Erstellen des Workflows verwendet wurden.

Lebenszyklus eines Blueprints

Blueprints werden entwickelt, getestet und registriert bei AWS Glue, und ausführen, um Workflows zu erstellen. Am Lebenszyklus von Blueprints sind in der Regel drei Personas beteiligt.

Persona	Aufgaben
AWS Glue Developer	• Schreibt das Workflow-Layoutskript und erstellt die Konfigurationsdatei. • Testet den Blueprint lokal mithilfe von Bibliotheken, die von AWS Glue Dienst. • Erstellt ein ZIP-Archiv des Skripts, der Konfigura tionsdatei und der unterstützenden Dateien und veröffentlicht das Archiv an einem Speicherort in Amazon S3. • Fügt dem Amazon S3 S3-Bucket eine Bucket-Richtlinie hinzu, die Leserechte für Bucket-Objekte gewährt AWS Glue AWS Administratorkonto. • Gewährt IAM-Leseberechtigungen für das ZIP-Archiv in Amazon S3 an AWS Glue Administrator.
AWS Glue Administrator	• Registriert den Blueprint bei AWS Glue. AWS Glue erstellt eine Kopie des ZIP-Archivs an einem reservier ten Amazon S3 S3-Speicherort.

Persona	Aufgaben
	• Gewährt Datenanalysten IAM-Berechtigungen für den Blueprint.
Datenanalyst	• Führt den Blueprint aus, um einen Workflow zu erstellen, und stellt die Blueprint-Parameterwerte bereit. Überprüft den Status der Blueprint-Ausführung, um sicherzustellen, dass der Workflow und die Workflow-Komponenten erfolgreich generiert wurden. • Führt den Workflow aus und ist für die Fehlerbeh ebung zuständig. Bevor Sie den Workflow ausführen, können Sie den Workflow überprüfen, indem Sie sich das Workflow-Entwurfsdiagramm auf der AWS Glue console.

Weitere Informationen finden Sie auch unter

- Entwicklung von Bauplänen in AWS Glue
- Erstellen eines Workflows aus einem Blueprint in AWS Glue
- Berechtigungen für Personas und Rollen für AWS Glue-Blueprints

Entwicklung von Bauplänen in AWS Glue

Als AWS Glue Entwickler können Blueprints erstellen und veröffentlichen, die Datenanalysten zur Generierung von Workflows verwenden können.

Themen

- Übersicht über die Entwicklung von Blueprints
- Voraussetzungen für die Entwicklung von Blueprints
- Schreiben des Blueprint-Codes
- Blueprint-Beispielprojekt
- Testen eines Blueprints
- Veröffentlichen eines Blueprints

- AWS Glue Referenz zu Blueprint-Klassen
- Blueprint-Beispiele

> Weitere Informationen finden Sie auch unter
>
> - Überblick über Baupläne in AWS Glue

Übersicht über die Entwicklung von Blueprints

Der erste Schritt im Entwicklungsprozess besteht darin, einen allgemeinen Anwendungsfall zu identifizieren, der von einem Blueprint profitieren würde. Ein typischer Anwendungsfall umfasst ein wiederkehrendes ETL-Problem, das auf allgemeine Weise gelöst werden sollte. Entwerfen Sie als Nächstes einen Blueprint, der den generalisierten Anwendungsfall implementiert, und definieren Sie die Blueprint-Eingabeparameter, die gemeinsam aus dem generalisierten Anwendungsfall einen spezifischen Anwendungsfall definieren können.

Ein Blueprint besteht aus einem Projekt, das eine Blueprint-Parameter-Konfigurationsdatei und ein Skript enthält, das das Layout des zu generierenden Workflows festlegt. Das Layout definiert die Aufträge und Crawler (oder Entitäten in Blueprint-Skript-Terminologie).

Sie geben im Layoutskript keine Auslöser direkt an. Stattdessen schreiben Sie Code, um die Abhängigkeiten zwischen den Jobs und Crawlern zu spezifizieren, die das Skript erstellt. AWS Glue generiert die Trigger auf der Grundlage Ihrer Abhängigkeitsspezifikationen. Die Ausgabe des Layoutskripts ist ein Workflow-Objekt, das Spezifikationen für alle Workflowentitäten enthält.

Sie erstellen Ihr Workflow-Objekt wie folgt AWS Glue Blueprint-Bibliotheken:

- `awsglue.blueprint.base_resource` – Eine Bibliothek von Basisressourcen, die von den Bibliotheken verwendet werden.
- `awsglue.blueprint.workflow` – Eine Bibliothek zum Definieren einer `Workflow`-Klasse.
- `awsglue.blueprint.job` – Eine Bibliothek zum Definieren einer `Job`-Klasse.
- `awsglue.blueprint.crawler` – Eine Bibliothek zum Definieren einer `Crawler`-Klasse.

Die einzigen anderen Bibliotheken, die zur Layoutgenerierung unterstützt werden, sind die für die Python-Shell verfügbaren Bibliotheken.

Bevor Sie Ihren Blueprint veröffentlichen, können Sie die in den Blueprint-Bibliotheken definierten Methoden verwenden, um den Blueprint lokal zu testen.

Wenn Sie bereit sind, den Blueprint für Datenanalysten verfügbar zu machen, verpacken Sie das Skript, die Parameterkonfigurationsdatei und alle unterstützenden Dateien, wie z. B. zusätzliche Skripts und Bibliotheken, in eine einzige bereitstellbare Komponente. Anschließend laden Sie das Asset auf Amazon S3 hoch und bitten einen Administrator, es zu registrieren bei AWS Glue.

Informationen zu weiteren Blueprint-Beispielprojekten finden Sie unter Blueprint-Beispielprojekt und Blueprint-Beispiele.

Voraussetzungen für die Entwicklung von Blueprints

Um Blueprints zu entwickeln, sollten Sie mit der Verwendung von AWS Glue und dem Schreiben von Skripten für Apache-Spark-ETL- oder Python-Shell-Aufträge vertraut sein. Außerdem müssen Sie die folgenden Einrichtungsaufgaben abschließen.

- Laden Sie vier AWS Python-Bibliotheken herunter, um sie in Ihren Blueprint-Layout-Skripten zu verwenden.
- Richten Sie das AWS SDKs ein.
- Richten Sie das ein AWS CLI.

Herunterladen der Python-Bibliotheken

Laden Sie die folgenden Bibliotheken von GitHub herunter und installieren Sie sie in Ihrem Projekt:

- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/base_resource.py
- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/workflow.py
- https://github.com/awslabs/aws-glue-blueprint-libs/.py tree/master/awsglue/blueprint/crawler
- https://github.com/awslabs/aws-glue-blueprint-libs/.py tree/master/awsglue/blueprint/job

Richten Sie das AWS Java-SDK ein

Für das AWS Java-SDK müssen Sie eine `jar` Datei hinzufügen, die die API für Blueprints enthält.

1. Falls Sie dies noch nicht getan haben, richten Sie das AWS SDK for Java ein.
   - Befolgen Sie für Java 1.x die Anweisungen unter Einrichten des AWS SDK für Java im AWS SDK für Java -Entwicklerhandbuch.

- Befolgen Sie für Java 2.x die Anweisungen unter Einrichten des AWS SDK for Java 2.x im AWS SDK for Java 2.x -Entwicklerhandbuch.

2. Laden Sie die `jar` Client-Datei herunter, die Zugriff auf die APIs für Blueprints hat.

   - Für Java 1.x: s3://awsglue-custom-blueprints-preview- -1.11.x.jar artifacts/awsglue-java-sdk-preview/AWSGlueJavaClient
   - Für Java 2.x: s3://awsglue-custom-blueprints-preview- artifacts/awsglue-java-sdk-v2-preview/AwsJavaSdk -Glue-2.0.jar

3. Fügen Sie den Client `jar` an den Anfang des Java-Klassenpfads, um den vom Java-SDK bereitgestellten AWS Glue-Client zu überschreiben. AWS

```
export CLASSPATH=<path-to-preview-client-jar>:$CLASSPATH
```

4. (Optional) Testen Sie das SDK mit der folgenden Java-Anwendung. Die Anwendung sollte eine leere Liste ausgeben.

   Ersetzen Sie `accessKey` und `secretKey` mit Ihren Anmeldeinformationen und ersetzen Sie `us-east-1` mit Ihrer Region.

```
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSCredentialsProvider;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.glue.AWSGlue;
import com.amazonaws.services.glue.AWSGlueClientBuilder;
import com.amazonaws.services.glue.model.ListBlueprintsRequest;

public class App{
    public static void main(String[] args) {
        AWSCredentials credentials = new BasicAWSCredentials("accessKey",
 "secretKey");
        AWSCredentialsProvider provider = new
 AWSStaticCredentialsProvider(credentials);
        AWSGlue glue = AWSGlueClientBuilder.standard().withCredentials(provider)
                .withRegion("us-east-1").build();
        ListBlueprintsRequest request = new
 ListBlueprintsRequest().withMaxResults(2);
        System.out.println(glue.listBlueprints(request));
    }
}
```

Richten Sie das AWS Python-SDK ein

Bei den folgenden Schritten wird davon ausgegangen, dass Sie Python Version 2.7 oder höher oder Version 3.9 oder höher auf Ihrem Computer installiert haben.

1. Laden Sie die folgende boto3-Wheel-Datei herunter. Wenn Sie zum Öffnen oder Speichern aufgefordert werden, speichern Sie die Datei. s3://awsglue-custom-blueprints-preview- artifacts/aws-python-sdk-preview/boto 3-1.17.31-py2.py3-none-any.whl
2. Laden Sie die folgende Botocore-Raddatei awsglue-custom-blueprints-preview herunter: artifacts/aws-python-sdk-preview/botocore s3://- -1.20.31-py2.py3-none-any.whl
3. Überprüfen Sie Ihre Python-Version.

   ```
   python --version
   ```

4. Geben Sie je nach Python-Version die folgenden Befehle ein (für Linux):

   - Für Python 2.7 oder höher.

     ```
     python3 -m pip install --user virtualenv
     source env/bin/activate
     ```

   - Für Python 3.9 oder höher.

     ```
     python3 -m venv python-sdk-test
     source python-sdk-test/bin/activate
     ```

5. Installieren Sie die Botocore-Wheel-Datei.

   ```
   python3 -m pip install <download-directory>/botocore-1.20.31-py2.py3-none-any.whl
   ```

6. Installieren Sie die boto3-Wheel-Datei.

   ```
   python3 -m pip install <download-directory>/boto3-1.17.31-py2.py3-none-any.whl
   ```

7. Konfigurieren Sie Ihre Anmeldeinformationen und Standardregion in den Dateien `~/.aws/credentials` und`~/.aws/config`. Weitere Informationen finden Sie unter Konfigurieren der AWS CLI im AWS Command Line Interface -Leitfaden.
8. (Optional) Testen Sie Ihre Einrichtung. Die folgenden Befehle sollten eine leere Liste zurückgeben.

Ersetzen Sie `us-east-1` durch Ihre Region.

```
$ python
>>> import boto3
>>> glue = boto3.client('glue', 'us-east-1')
>>> glue.list_blueprints()
```

Richten Sie die Vorschau ein AWS CLI

1. Falls Sie dies noch nicht getan haben, installieren und/oder aktualisieren Sie AWS Command Line Interface (AWS CLI) auf Ihrem Computer. Am einfachsten geht dies über `pip`, das Python-Installationsdienstprogramm:

   ```
   pip install awscli --upgrade --user
   ```

   Ausführliche Installationsanleitungen für die AWS CLI finden Sie unter Installieren der AWS Command Line Interface.
2. Laden Sie die AWS CLI Raddatei von: s3://awsglue-custom-blueprints-preview- artifacts/awscli-preview-build/awscli -1.19.31-py2.py3-none-any.whl herunter
3. Installieren Sie die AWS CLI Raddatei.

   ```
   python3 -m pip install awscli-1.19.31-py2.py3-none-any.whl
   ```
4. Führen Sie den Befehl `aws configure` aus. Konfigurieren Sie Ihre AWS Anmeldeinformationen (einschließlich Zugriffsschlüssel und geheimer Schlüssel) und AWS Region. Informationen zur Konfiguration von finden Sie AWS CLI hier: Konfiguration der AWS CLI.
5. Testen Sie die AWS CLI. Der folgende Befehl sollte eine leere Liste zurückgeben.

   Ersetzen Sie `us-east-1` durch Ihre Region.

   ```
   aws glue list-blueprints --region us-east-1
   ```

Schreiben des Blueprint-Codes

Jedes von Ihnen erstellte Blueprint-Projekt muss mindestens die folgenden Dateien enthalten:

- Ein Python-Layoutskript, das den Workflow definiert. Das Skript enthält eine Funktion, die die Entitäten (Aufträge und Crawler) in einem Workflow und die Abhängigkeiten zwischen ihnen definiert.
- Eine Konfigurationsdatei, `blueprint.cfg`, die Folgendes definiert:
  - Den vollständigen Pfad der Workflow-Layoutdefinitionsfunktion.
  - Die Parameter, die der Blueprint akzeptiert.

Themen

- Erstellen des Blueprint-Layoutskripts
- Erstellen der Konfigurationsdatei
- Angeben der Blueprint-Parameter

Erstellen des Blueprint-Layoutskripts

Das Blueprint-Layoutskript muss eine Funktion enthalten, die die Entitäten in Ihrem Workflow generiert. Sie können diese Funktion benennen, wie Sie möchten. AWS Glue verwendet die Konfigurationsdatei, um den vollständig qualifizierten Namen der Funktion zu ermitteln.

Die Layoutfunktion führt Folgendes durch:

- (Optional) Instanziiert die `Job`-Klasse, um `Job`-Objekte zu erstellen, und übergibt Argumente wie `Command` und `Role`. Dies sind Jobeigenschaften, die Sie angeben würden, wenn Sie den Job mit dem AWS Glue Konsole oder API.
- (Optional) Instanziiert die `Crawler`-Klasse, um `Crawler`-Objekte zu erstellen, und übergibt Argumente wie Name, Rolle und Ziel.
- Übergibt die zusätzlichen Argumente `DependsOn` und `WaitForDependencies` an `Job()` und `Crawler()`, um Abhängigkeiten zwischen den Objekten (Workflow-Entitäten) anzugeben. Diese Argumente werden später in diesem Abschnitt erläutert.
- Instanziiert die `Workflow` Klasse, um das Workflow-Objekt zu erstellen, an das zurückgegeben wird AWS Glue, übergibt ein `Name` Argument, ein `Entities` Argument und ein optionales `OnSchedule` Argument. Das `Entities`-Argument gibt alle Aufträge und Crawler an, die im Workflow enthalten sein sollen. Um zu erfahren, wie man ein `Entities`-Objekt aufbaut, sehen Sie sich das Beispielprojekt weiter unten in diesem Abschnitt an.
- Gibt das `Workflow`-Objekt zurück.

Definitionen der Klassen `Job`, `Crawler` und `Workflow` finden Sie unter AWS Glue Referenz zu Blueprint-Klassen.

Die Layoutfunktion muss die folgenden Eingabeargumente akzeptieren.

Argument	Beschreibung
`user_params`	Python-Wörterbuch der Blueprint-Parameternamen und -werte. Weitere Informationen finden Sie unter Angeben der Blueprint-Parameter.
`system_params`	Python-Wörterbuch, das zwei Eigenschaften enthält: `region` und `accountId` .

Hier ist ein Layoutgenerator-Beispielskript in einer Datei namens `Layout.py`:

```
import argparse
import sys
import os
import json
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *


def generate_layout(user_params, system_params):

    etl_job = Job(Name="{}_etl_job".format(user_params['WorkflowName']),
                  Command={
                      "Name": "glueetl",
                      "ScriptLocation": user_params['ScriptLocation'],
                      "PythonVersion": "2"
                  },
                  Role=user_params['PassRole'])
    post_process_job = Job(Name="{}_post_process".format(user_params['WorkflowName']),
                           Command={
                               "Name": "pythonshell",
                               "ScriptLocation": user_params['ScriptLocation'],
                               "PythonVersion": "2"
                           },
                           Role=user_params['PassRole'],
                           DependsOn={
```

```
                            etl_job: "SUCCEEDED"
                        },
                        WaitForDependencies="AND")
    sample_workflow = Workflow(Name=user_params['WorkflowName'],
                        Entities=Entities(Jobs=[etl_job, post_process_job]))
    return sample_workflow
```

Das Beispielskript importiert die erforderlichen Blueprint-Bibliotheken und enthält eine `generate_layout`-Funktion, die einen Workflow mit zwei Aufträgen generiert. Dies ist ein sehr einfaches Skript. Ein komplexeres Skript könnte zusätzliche Logik und Parameter verwenden, um einen Workflow mit vielen Aufträgen und Crawlern oder sogar einer variablen Anzahl von Aufträgen und Crawlern zu generieren.

Verwenden des DependsOn Arguments

Das `DependsOn`-Argument ist eine Wörterbuchdarstellung einer Abhängigkeit dieser Entität von anderen Entitäten innerhalb des Workflows. Es hat das folgende Format.

```
DependsOn = {dependency1 : state, dependency2 : state, ...}
```

Die Schlüssel in diesem Wörterbuch stellen die Objektreferenz und nicht den Namen der Entität dar, während es sich bei den Werten um Zeichenketten handelt, die dem Zustand entsprechen, auf den Sie achten müssen. AWS Glue leitet die richtigen Auslöser ab. Informationen zu den gültigen Zuständen finden Sie unter Bedingungsstruktur.

Beispielsweise kann ein Auftrag vom erfolgreichen Abschluss eines Crawlers abhängen. Wenn Sie ein Crawler-Objekt namens `crawler2` wie folgt festlegen:

```
crawler2 = Crawler(Name="my_crawler", ...)
```

Dann enthält ein Objekt, das von `crawler2` abhängig ist, ein Konstruktor-Argument wie folgt:

```
DependsOn = {crawler2 : "SUCCEEDED"}
```

Zum Beispiel:

```
job1 = Job(Name="Job1", ..., DependsOn = {crawler2 : "SUCCEEDED", ...})
```

Wenn `DependsOn` für eine Entität weggelassen wird, hängt diese Entität vom Workflow-Startauslöser ab.

Das WaitForDependencies Argument verwenden

Das `WaitForDependencies`-Argument definiert, ob eine Auftrags- oder Crawler-Entität warten soll, bis alle Entitäten, von denen sie abhängig ist, abgeschlossen sind, oder bis irgendeine abgeschlossen ist.

Die zulässigen Werte sind „AND“ oder „ANY“.

Verwenden des OnSchedule Arguments

Das `OnSchedule`-Argument für den `Workflow`-Klassenkonstruktor ist ein `cron`-Ausdruck, der die Startauslöserdefinition für einen Workflow definiert.

Wenn dieses Argument angegeben ist, AWS Glue erstellt einen Zeitplan-Trigger mit dem entsprechenden Zeitplan. Wenn es nicht angegeben wird, ist der Startauslöser für den Workflow ein On-Demand-Auslöser.

Erstellen der Konfigurationsdatei

Die Blueprint-Konfigurationsdatei ist eine erforderliche Datei, die den Skripteintrittspunkt zum Generieren des Workflows und die Parameter definiert, die der Blueprint akzeptiert. Die Datei muss mit `blueprint.cfg` benannt werden.

Hier ist eine Beispielkonfigurationsdatei.

```
{
    "layoutGenerator": "DemoBlueprintProject.Layout.generate_layout",
    "parameterSpec" : {
           "WorkflowName" : {
                "type": "String",
                "collection": false
           },
           "WorkerType" : {
                "type": "String",
                "collection": false,
                "allowedValues": ["G1.X", "G2.X"],
                "defaultValue": "G1.X"
           },
           "Dpu" : {
                "type" : "Integer",
                "allowedValues" : [2, 4, 6],
                "defaultValue" : 2
```

```
            },
            "DynamoDBTableName": {
                 "type": "String",
                 "collection" : false
            },
            "ScriptLocation" : {
                 "type": "String",
                 "collection": false
     }
    }
}
```

Die `layoutGenerator`-Eigenschaft gibt den vollständig qualifizierten Namen der Funktion in dem Skript an, das das Layout generiert.

Die `parameterSpec`-Eigenschaft gibt die Parameter an, die dieser Blueprint akzeptiert. Weitere Informationen finden Sie unter Angeben der Blueprint-Parameter.

> **Important**
>
> Ihre Konfigurationsdatei muss den Workflow-Namen als Blueprint-Parameter enthalten oder Sie müssen einen eindeutigen Workflow-Namen in Ihrem Layoutskript generieren.

Angeben der Blueprint-Parameter

Die Konfigurationsdatei enthält Blueprint-Parameterspezifikationen in einem `parameterSpec`-JSON-Objekt. `parameterSpec` enthält ein oder mehrere Parameterobjekte.

```
"parameterSpec": {
    "<parameter_name>": {
      "type": "<parameter-type>",
      "collection": true|false,
      "description": "<parameter-description>",
      "defaultValue": "<default value for the parameter if value not specified>"
      "allowedValues": "<list of allowed values>"
    },
    "<parameter_name>": {
       ...
    }
  }
```

Im Folgenden sind die Regeln für die Codierung jedes Parameterobjekts aufgeführt:

- Der Parametername und `type` sind obligatorisch. Alle anderen Eigenschaften sind optional.
- Wenn Sie die `defaultValue`-Eigenschaft angeben, ist der Parameter optional. Anderenfalls ist der Parameter obligatorisch und der Datenanalyst, der einen Workflow aus dem Blueprint erstellt, muss dafür einen Wert angeben.
- Wenn Sie die `collection`-Eigenschaft auf `true` festlegen, kann der Parameter eine Sammlung von Werten annehmen. Sammlungen können von einem beliebigen Datentyp sein.
- Wenn Sie angeben`allowedValues`, AWS Glue Die Konsole zeigt eine Dropdownliste mit Werten an, aus denen der Datenanalyst wählen kann, wenn er einen Workflow anhand des Blueprints erstellt.

Die folgenden Werte sind für `type` zulässig:

Parameterdatentyp	Hinweise
`String`	-
`Integer`	-
`Double`	-
`Boolean`	Mögliche Werte sind `true` und `false`. Generiert ein Kontrollkästchen auf der <blueprint>Seite „Workflow erstellen von“ auf der AWS Glue console.
`S3Uri`	Vollständiger Amazon-S3-Pfad, beginnend mit `s3://`. Generiert ein Textfeld und eine Browse (Durchsuchen)-Schaltfläche auf der Seite Create a workflow from <blueprint> (Workflow aus <Blueprint> erstellen).
`S3Bucket`	Nur Amazon-S3-Bucket-Name. Generiert eine Bucket-Auswahl auf der Seite Create a workflow from <blueprint> (Workflow aus <Blueprint> erstellen).
`IAMRoleArn`	Amazon-Ressourcenname (ARN) einer AWS Identity and Access Management (IAM) -Rolle. Generiert eine Rollenauswahl auf der Seite Create a workflow from <blueprint> (Workflow aus <Blueprint> erstellen).

Parameterdatentyp	Hinweise
`IAMRoleName`	Name einer IAM-Rolle. Generiert eine Rollenauswahl auf der Seite Create a workflow from <blueprint> (Workflow aus <Blueprint> erstellen).

Blueprint-Beispielprojekt

Die Datenformatkonvertierung ist ein häufiger Anwendungsfall für Extract, Transform, Load (ETL). In typischen analytischen Workloads werden spaltenbasierte Dateiformate wie Parquet und ORC gegenüber Textformaten wie CSV und JSON bevorzugt. Mit diesem Beispiel-Blueprint können Sie Daten CSV/JSON/etc von. in Parquet für Dateien auf Amazon S3 konvertieren.

Dieser Blueprint verwendet eine Liste von S3-Pfaden, die durch einen Blueprint-Parameter definiert sind, konvertiert die Daten ins Parquet-Format und schreibt sie in den von einem anderen Blueprint-Parameter angegebenen S3-Speicherort. Das Layoutskript erstellt einen Crawler und einen Auftrag für jeden Pfad. Das Layoutskript lädt auch das ETL-Skript in `Conversion.py` in einen S3-Bucket hoch, der durch einen anderen Blueprint-Parameter angegeben wird. Das Layoutskript gibt dann das hochgeladene Skript als ETL-Skript für jeden Auftrag an. Das ZIP-Archiv für das Projekt enthält das Layoutskript, das ETL-Skript und die Blueprint-Konfigurationsdatei.

Informationen zu weiteren Blueprint-Beispielprojekten finden Sie unter Blueprint-Beispiele.

Nachfolgend sehen Sie das Layoutskript in der Datei `Layout.py`.

```
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *
import boto3

s3_client = boto3.client('s3')

# Ingesting all the S3 paths as Glue table in parquet format
def generate_layout(user_params, system_params):
    #Always give the full path for the file
    with open("ConversionBlueprint/Conversion.py", "rb") as f:
        s3_client.upload_fileobj(f, user_params['ScriptsBucket'], "Conversion.py")
    etlScriptLocation = "s3://{}/Conversion.py".format(user_params['ScriptsBucket'])

    crawlers = []
    jobs = []
```

```
    workflowName = user_params['WorkflowName']
    for path in user_params['S3Paths']:
      tablePrefix = "source_"
      crawler = Crawler(Name="{}_crawler".format(workflowName),
                        Role=user_params['PassRole'],
                        DatabaseName=user_params['TargetDatabase'],
                        TablePrefix=tablePrefix,
                        Targets= {"S3Targets": [{"Path": path}]})
      crawlers.append(crawler)
      transform_job = Job(Name="{}_transform_job".format(workflowName),
                          Command={"Name": "glueetl",
                                   "ScriptLocation": etlScriptLocation,
                                   "PythonVersion": "3"},
                          Role=user_params['PassRole'],
                          DefaultArguments={"--database_name":
 user_params['TargetDatabase'],
                                            "--table_prefix": tablePrefix,
                                            "--region_name": system_params['region'],
                                            "--output_path":
 user_params['TargetS3Location']},
                          DependsOn={crawler: "SUCCEEDED"},
                          WaitForDependencies="AND")
      jobs.append(transform_job)
    conversion_workflow = Workflow(Name=workflowName, Entities=Entities(Jobs=jobs,
 Crawlers=crawlers))
    return conversion_workflow
```

Nachfolgend sehen Sie die entsprechende Blueprint-Konfigurationsdatei `blueprint.cfg`.

```
{
    "layoutGenerator": "ConversionBlueprint.Layout.generate_layout",
    "parameterSpec" : {
        "WorkflowName" : {
            "type": "String",
            "collection": false,
            "description": "Name for the workflow."
        },
        "S3Paths" : {
            "type": "S3Uri",
            "collection": true,
            "description": "List of Amazon S3 paths for data ingestion."
        },
        "PassRole" : {
```

```
            "type": "IAMRoleName",
            "collection": false,
            "description": "Choose an IAM role to be used in running the job/crawler"
        },
        "TargetDatabase": {
            "type": "String",
            "collection" : false,
            "description": "Choose a database in the Data Catalog."
        },
        "TargetS3Location": {
            "type": "S3Uri",
            "collection" : false,
            "description": "Choose an Amazon S3 output path: ex:s3://<target_path>/."
        },
        "ScriptsBucket": {
            "type": "S3Bucket",
            "collection": false,
            "description": "Provide an S3 bucket name(in the same AWS Region) to store
 the scripts."
        }
    }
}
```

Das folgende Skript in der Datei `Conversion.py` ist das hochgeladene ETL-Skript. Beachten Sie, dass das Partitionierungsschema während der Konvertierung beibehalten wird.

```
import sys
from pyspark.sql.functions import *
from pyspark.context import SparkContext
from awsglue.transforms import *
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions
import boto3

args = getResolvedOptions(sys.argv, [
    'JOB_NAME',
    'region_name',
    'database_name',
    'table_prefix',
    'output_path'])
databaseName = args['database_name']
tablePrefix = args['table_prefix']
```

```
outputPath = args['output_path']

glue = boto3.client('glue', region_name=args['region_name'])

glue_context = GlueContext(SparkContext.getOrCreate())
spark = glue_context.spark_session
job = Job(glue_context)
job.init(args['JOB_NAME'], args)

def get_tables(database_name, table_prefix):
    tables = []
    paginator = glue.get_paginator('get_tables')
    for page in paginator.paginate(DatabaseName=database_name, Expression=table_prefix
+"*"):
        tables.extend(page['TableList'])
    return tables

for table in get_tables(databaseName, tablePrefix):
    tableName = table['Name']
    partitionList = table['PartitionKeys']
    partitionKeys = []
    for partition in partitionList:
        partitionKeys.append(partition['Name'])

    # Create DynamicFrame from Catalog
    dyf = glue_context.create_dynamic_frame.from_catalog(
        name_space=databaseName,
        table_name=tableName,
        additional_options={
            'useS3ListImplementation': True
        },
        transformation_ctx='dyf'
    )

    # Resolve choice type with make_struct
    dyf = ResolveChoice.apply(
        frame=dyf,
        choice='make_struct',
        transformation_ctx='resolvechoice_' + tableName
    )

    # Drop null fields
    dyf = DropNullFields.apply(
        frame=dyf,
```

```
        transformation_ctx="dropnullfields_" + tableName
    )

    # Write DynamicFrame to S3 in glueparquet
    sink = glue_context.getSink(
        connection_type="s3",
        path=outputPath,
        enableUpdateCatalog=True,
        partitionKeys=partitionKeys
    )
    sink.setFormat("glueparquet")

    sink.setCatalogInfo(
        catalogDatabase=databaseName,
        catalogTableName=tableName[len(tablePrefix):]
    )
    sink.writeFrame(dyf)

job.commit()
```

> **Note**
>
> Nur zwei Amazon-S3-Pfade können als Eingabe für den Beispiel-Blueprint bereitgestellt werden. Das liegt daran AWS Glue Trigger sind darauf beschränkt, nur zwei Crawler-Aktionen aufzurufen.

Testen eines Blueprints

Während Sie Ihren Code entwickeln, sollten Sie lokale Tests durchführen, um zu überprüfen, ob das Workflow-Layout korrekt ist.

Lokales Testen generiert nicht AWS Glue Jobs, Crawler oder Trigger. Stattdessen führen Sie das Layoutskript lokal aus und verwenden die Methoden `to_json()` und`validate()`, um Objekte zu drucken und Fehler zu finden. Diese Methoden sind in allen drei Klassen verfügbar, die in den Bibliotheken definiert sind.

Es gibt zwei Möglichkeiten, mit den `system_params` Argumenten `user_params` und umzugehen AWS Glue geht zu Ihrer Layoutfunktion über. Ihr Testbench-Code kann ein Wörterbuch mit Beispiel-Blueprint-Parameterwerten erstellen und dieses als `user_params`-Argument an die Layoutfunktion

übergeben. Sie können auch die Verweise auf `user_params` entfernen und durch hartcodierte Zeichenfolgen ersetzen.

Wenn Ihr Code die Eigenschaften `region` und `accountId` im Argument `system_params` verwendet, können Sie Ihr eigenes Wörterbuch für `system_params` übergeben.

Blueprint testen

1. Starten Sie einen Python-Interpreter in einem Verzeichnis mit den Bibliotheken oder laden Sie die Blueprint-Dateien und die bereitgestellten Bibliotheken in Ihre bevorzugte integrierte Entwicklungsumgebung (IDE).
2. Stellen Sie sicher, dass Ihr Code die bereitgestellten Bibliotheken importiert.
3. Fügen Sie Code zu Ihrer Layoutfunktion hinzu, um `validate()` oder `to_json()` für eine beliebige Entität oder das `Workflow`-Objekt aufzurufen. Wenn Ihr Code beispielsweise ein `Crawler`-Objekt namens `mycrawler` erstellt, können Sie `validate()` wie folgt aufrufen.

   ```
   mycrawler.validate()
   ```

   Sie können `mycrawler` wie folgt drucken:

   ```
   print(mycrawler.to_json())
   ```

   Wenn Sie `to_json` für ein Objekt aufrufen, müssen Sie `validate()` nicht auch aufrufen, da `to_json()` `validate()` aufruft.

   Am nützlichsten ist es, diese Methoden für das Workflow-Objekt aufzurufen. Angenommen, das Skript benennt das Workflowobjekt mit `my_workflow`, validieren und drucken Sie das Workflow-Objekt wie folgt.

   ```
   print(my_workflow.to_json())
   ```

   Weitere Informationen zu `to_json()` und `validate()` finden Sie unter Klassenmethoden.

   Sie können auch `pprint` importieren und das Workflow-Objekt im Schöndruck drucken, wie im Beispiel weiter unten in diesem Abschnitt gezeigt.
4. Führen Sie den Code aus, beheben Sie Fehler und entfernen Sie schließlich alle Aufrufe von `validate()` oder `to_json()`.

Example

Das folgende Beispiel zeigt, wie ein Wörterbuch mit Blueprint-Beispielparametern erstellt und als `user_params`-Argument an die Layoutfunktion `generate_compaction_workflow` übergeben wird. Außerdem wird gezeigt, wie das generierte Workflow-Objekt im Schöndruck gedruckt wird.

```
from pprint import pprint
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *

USER_PARAMS = {"WorkflowName": "compaction_workflow",
               "ScriptLocation": "s3://amzn-s3-demo-bucket/scripts/threaded-
compaction.py",
               "PassRole": "arn:aws:iam::111122223333:role/GlueRole-ETL",
               "DatabaseName": "cloudtrial",
               "TableName": "ct_cloudtrail",
               "CoalesceFactor": 4,
               "MaxThreadWorkers": 200}


def generate_compaction_workflow(user_params: dict, system_params: dict) -> Workflow:
    compaction_job = Job(Name=f"{user_params['WorkflowName']}_etl_job",
                         Command={"Name": "glueetl",
                                  "ScriptLocation": user_params['ScriptLocation'],
                                  "PythonVersion": "3"},
                         Role="arn:aws:iam::111122223333:role/
AWSGlueServiceRoleDefault",
                         DefaultArguments={"DatabaseName": user_params['DatabaseName'],
                                           "TableName": user_params['TableName'],
                                           "CoalesceFactor":
 user_params['CoalesceFactor'],
                                           "max_thread_workers":
 user_params['MaxThreadWorkers']})

    catalog_target = {"CatalogTargets": [{"DatabaseName": user_params['DatabaseName'],
 "Tables": [user_params['TableName']]}]}

    compacted_files_crawler = Crawler(Name=f"{user_params['WorkflowName']}_post_crawl",
                                      Targets = catalog_target,
                                      Role=user_params['PassRole'],
                                      DependsOn={compaction_job: "SUCCEEDED"},
                                      WaitForDependencies="AND",
```

```
                                    SchemaChangePolicy={"DeleteBehavior": "LOG"})

    compaction_workflow = Workflow(Name=user_params['WorkflowName'],
                                   Entities=Entities(Jobs=[compaction_job],

 Crawlers=[compacted_files_crawler]))
    return compaction_workflow

generated = generate_compaction_workflow(user_params=USER_PARAMS, system_params={})
gen_dict = generated.to_json()

pprint(gen_dict)
```

Veröffentlichen eines Blueprints

Nachdem Sie einen Blueprint entwickelt haben, müssen Sie ihn in Amazon S3 hochladen. Sie müssen Schreibberechtigungen für den Amazon-S3-Bucket haben, der zum Veröffentlichen des Blueprints verwendet wird. Sie müssen auch sicherstellen, dass der AWS Glue-Administrator, der den Blueprint registrieren wird, über Lesezugriff auf den Amazon-S3-Bucket verfügt. Die vorgeschlagenen AWS Identity and Access Management (IAM-) Berechtigungsrichtlinien für Personas und Rollen für AWS Glue Blueprints finden Sie unter. Berechtigungen für Personas und Rollen für AWS Glue-Blueprints

Blueprint veröffentlichen

1. Erstellen Sie die erforderlichen Skripte und Ressourcen und die Blueprint-Konfigurationsdatei.
2. Fügen Sie alle Dateien zu einem ZIP-Archiv hinzu und laden Sie die ZIP-Datei in Amazon S3 hoch. Verwenden Sie einen S3 Bucket, der sich in der Region befindet, in der Benutzer den Blueprint registrieren und ausführen werden.

   Sie können über die Befehlszeile mit dem folgenden Befehl eine ZIP-Datei erstellen.

   ```
   zip -r folder.zip folder
   ```
3. Fügen Sie eine Bucket-Richtlinie hinzu, die dem gewünschten Konto Leseberechtigungen gewährt. AWS Im Folgenden finden Sie eine Beispielrichtlinie.

   JSON

   ```
   {
   ```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-blueprints/*"
        }
    ]
}
```

4. Gewähren Sie dem AWS Glue-Administrator oder der Person, die Blueprints registrieren wird, die IAM-Berechtigung `s3:GetObject` für den Amazon-S3-Bucket. Eine Beispielrichtlinie zur Erteilung an Administratoren finden Sie unter AWS Glue-Administratorberechtigungen für Blueprints.

Nachdem Sie das lokale Testen Ihres Blueprints abgeschlossen haben, könnten Sie einen Blueprint auch in AWS Glue testen. Um einen Blueprint in AWS Glue zu testen, muss er registriert sein. Sie können mithilfe von IAM-Autorisierung oder durch Verwendung separater Testkonten einschränken, wer den registrierten Blueprint sehen kann.

> Weitere Informationen finden Sie auch unter:
>
> - Registrierung eines Blueprints in AWS Glue

AWS Glue Referenz zu Blueprint-Klassen

Die Bibliotheken für AWS Glue Blueprints definieren drei Klassen, die Sie in Ihrem Workflow-Layout-Skript verwenden: `JobCrawler`, und`Workflow`.

Themen

- Auftragsklasse
- Crawler-Klasse
- Workflow-Klasse
- Klassenmethoden

Auftragsklasse

Die `Job` Klasse steht für AWS Glue ETL-Job.

Obligatorische Konstruktor-Argumente

Nachfolgend sind obligatorische Konstruktor-Argumente für die `Job`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`Name`	`str`	Name, der dem Job zugewiesen werden soll. AWS Glue fügt dem Namen ein zufällig generiertes Suffix hinzu, um den Job von denen zu unterscheiden, die durch andere Blueprint-Läufe erstellt wurden.
`Role`	`str`	Amazon-Ressourcenname (ARN) der Rolle, die der Auftrag bei der Ausführung übernehmen soll.
`Command`	`dict`	Auftragsbefehl, wie in JobCommand Struktur in der API-Dokumentation angegeben.

Optionale Konstruktor-Argumente

Nachfolgend sind optionale Konstruktor-Argumente für die `Job`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`DependsOn`	`dict`	Liste der Workflow-Entitäten, von denen der Auftrag abhängt. Weitere Informationen finden Sie unter Verwenden des DependsOn Arguments.
`WaitForDependencies`	`str`	Gibt an, ob der Auftrag warten soll, bis alle Entitäten, von denen er abhängig ist, vor der Ausführung abgeschlossen sind, oder bis irgendeine abgeschlossen ist. Weitere Informationen finden Sie unter Das WaitForDe

Argumentname	Typ	Beschreibung
		pendencies Argument verwenden. Lassen Sie das Argument weg, wenn der Auftrag nur von einer Entität abhängt.
(Auftragseigenschaften)	-	Jede der Auftragseigenschaften, die in Auftrags-Struktur der aufgeführt sind AWS Glue API-Dokumentation (außer `CreatedOn` und`LastModifiedOn`).

Crawler-Klasse

Die `Crawler` Klasse steht für AWS Glue Crawler.

Obligatorische Konstruktor-Argumente

Nachfolgend sind obligatorische Konstruktor-Argumente für die `Crawler`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`Name`	`str`	Name, der dem Crawler zugewiesen werden soll. AWS Glue fügt dem Namen ein zufällig generiertes Suffix hinzu, um den Crawler von denen zu unterscheiden, die durch andere Blueprint-Läufe erstellt wurden.
`Role`	`str`	ARN der Rolle, die der Crawler während der Ausführung übernehmen soll.
`Targets`	`dict`	Sammlung von Zielen für das Crawlen. `Targets`-Klassen-Konstruktor-Argumente sind in der CrawlerTargets Struktur in der API-Dokum entation definiert. Alle `Targets`-Konstruktor-Argumente sind optional, Sie müssen jedoch mindestens eines übergeben.

Optionale Konstruktor-Argumente

Nachfolgend sind optionale Konstruktor-Argumente für die `Crawler`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`DependsOn`	`dict`	Liste der Workflow-Entitäten, von denen der Crawler abhängt. Weitere Informationen finden Sie unter Verwenden des DependsOn Arguments.
`WaitForDependencies`	`str`	Gibt an, ob der Crawler warten soll, bis alle Entitäten, von denen er abhängig ist, vor der Ausführung abgeschlossen sind, oder bis irgendeine abgeschlossen ist. Weitere Informationen finden Sie unter Das WaitForDependencies Argument verwenden. Lassen Sie das Argument weg, wenn der Crawler nur von einer Entität abhängt.
(Crawler-Eigenschaften)	-	Alle Crawler-Eigenschaften, die in der aufgeführt sind Crawler-Struktur AWS Glue API-Dokumentation, mit den folgenden Ausnahmen: • `State` • `CrawlElapsedTime` • `CreationTime` • `LastUpdated` • `LastCrawl` • `Version`

Workflow-Klasse

Die `Workflow` Klasse steht für AWS Glue Arbeitsablauf. Das Workflow-Layout-Skript gibt ein `Workflow` Objekt zurück. AWS Glue erstellt einen Workflow, der auf diesem Objekt basiert.

Obligatorische Konstruktor-Argumente

Nachfolgend sind obligatorische Konstruktor-Argumente für die `Workflow`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`Name`	`str`	Name, der dem Workflow zugewiesen werden soll.
`Entities`	`Entities`	Eine Sammlung von Entitäten (Aufträgen und Crawlern), die im Workflow enthalten sein sollen. Der `Entities`-Klassen-Konstruktor akzeptier t ein `Jobs`-Argument, bei dem es sich um eine Liste von `Job`-Objekten handelt, und ein `Crawlers`-Argument, bei dem es sich um eine Liste von `Crawler`-Objekten handelt.

Optionale Konstruktor-Argumente

Nachfolgend sind optionale Konstruktor-Argumente für die `Workflow`-Klasse aufgeführt.

Argumentname	Typ	Beschreibung
`Description`	`str`	Siehe Workflow-Struktur.
`DefaultRunProperti es`	`dict`	Siehe Workflow-Struktur.
`OnSchedule`	`str`	Ein `cron`-Ausdruck.

Klassenmethoden

Alle drei Klassen umfassen die folgenden Methoden.

validate()

Validiert die Eigenschaften des Objekts und gibt eine Meldung aus und wird beendet, wenn Fehler gefunden werden. Generiert keine Ausgabe, wenn keine Fehler vorliegen. Ruft sich für die `Workflow`-Klasse selbst für jede Entität im Workflow auf.

to_json()

Serialisiert das Objekt in JSON. Ruft auch `validate()` auf. Für die `Workflow`-Klasse enthält das JSON-Objekt Auftrags- und Crawler-Listen sowie eine Liste von Auslösern, die von den Auftrags- und Crawler-Abhängigkeitsspezifikationen generiert werden.

Blueprint-Beispiele

Es gibt eine Reihe von Beispiel-Blueprint-Projekten auf der AWS Glue Blueprint Github-Repository. Diese Beispiele dienen nur als Referenz und sind nicht für die Produktion bestimmt.

Die Titel der Beispielprojekte lauten:

- Compaction (Komprimierung): Dieser Blueprint erstellt einen Auftrag, der Eingabedateien basierend auf der gewünschten Dateigröße in größere Blöcke komprimiert.
- Conversion (Konvertierung): Dieser Blueprint konvertiert Eingabedateien in verschiedenen Standard-Dateiformaten ins Apache-Parquet-Format, das für analytische Workloads optimiert ist.
- Crawling Amazon S3 locations (Crawling von Amazon-S3-Speicherorten): Dieser Blueprint durchsucht mehrere Amazon-S3-Speicherorte, um Metadatentabellen zum Data Catalog hinzuzufügen.
- Benutzerdefinierte Verbindung zum Datenkatalog: Dieser Blueprint greift auf Datenspeicher zu unter Verwendung von AWS Glue benutzerdefinierte Konnektoren, liest die Datensätze und füllt die Tabellendefinitionen im AWS Glue-Datenkatalog auf der Grundlage des Datensatzschemas.
- Encoding (Codierung): Dieser Blueprint konvertiert Nicht-UTF-Dateien in UTF-codierte Dateien.
- Partitioning (Partitionierung): Dieser Blueprint erstellt einen Partitionierungsauftrag, der Ausgabedateien basierend auf bestimmten Partitionsschlüsseln in Partitionen platziert.
- Importing Amazon S3 data into a DynamoDB table (Importieren von Amazon-S3-Daten in eine DynamoDB-Tabelle): Dieser Blueprint importiert Daten aus Amazon S3 in eine DynamoDB-Tabelle.
- Standardtabelle zur Steuerung: Dieser Blueprint importiert eine AWS Glue Data Catalog-Tabelle in eine Lake Formation-Tabelle.

Registrierung eines Blueprints in AWS Glue

Nach dem AWS Glue Der Entwickler hat den Blueprint codiert und ein ZIP-Archiv auf Amazon Simple Storage Service (Amazon S3) hochgeladen, ein AWS Glue Der Administrator muss den Blueprint registrieren. Die Registrierung sorgt dafür, dass der Blueprint verwendet werden kann.

Wenn Sie einen Blueprint registrieren, AWS Glue kopiert das Blueprint-Archiv an einen reservierten Amazon S3 S3-Speicherort. Anschließend können Sie das Archiv aus dem Upload-Speicherort löschen.

Um einen Blueprint zu registrieren, benötigen Sie Leseberechtigungen für den Amazon-S3-Speicherort, an dem sich das hochgeladene Archiv befindet. Sie benötigen außerdem die AWS Identity and Access Management (IAM-) Genehmigung. `glue:CreateBlueprint` Für die vorgeschlagenen Berechtigungen für eine AWS Glue Ein Administrator, der Blueprints registrieren, anzeigen und verwalten muss, finden Sie unterAWS Glue-Administratorberechtigungen für Blueprints.

Sie können einen Blueprint registrieren, indem Sie AWS Glue Konsole, AWS Glue API oder AWS Command Line Interface (AWS CLI).

Einen Blueprint registrieren (Konsole)

1. Stellen Sie sicher, dass Sie Leseberechtigungen (`s3:GetObject`) für das Blueprint-ZIP-Archiv in Amazon S3 haben.
2. Öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.

   Melden Sie sich als Benutzer mit Berechtigungen zum Registrieren eines Blueprints an. Wechseln Sie zur AWS -Region des Amazon-S3-Buckets, der das Blueprint-ZIP-Archiv enthält.
3. Wählen Sie im Navigationsbereich die Option Blueprints aus. Wählen Sie dann auf der Seite Blueprints die Option Add blueprint (Blueprint hinzufügen aus.
4. Geben Sie einen Namen und optional eine Beschreibung ein.
5. Geben Sie unter ZIP archive location (S3) (Speicherort des ZIP-Archivs (S3)) den Amazon-S3-Pfad des hochgeladenen Blueprint-ZIP-Archivs ein. Fügen Sie dabei den Namen der Archivdatei hinzu und beginnen Sie den Pfad mit `s3://`.
6. (Optional) Fügen Sie einen oder mehrere Tags hinzu.
7. Wählen Sie Add blueprint (Blueprint hinzufügen) aus.

   Die Seite Blueprints wird wieder geöffnet und zeigt den Blueprint-Status `CREATING` an. Wählen Sie die Schaltfläche „Refresh“ (Aktualisieren) aus, bis sich der Status in `ACTIVE` oder `FAILED` ändert.
8. Wenn der Status `FAILED` lautet, wählen Sie den Blueprint und dann im Menü Actions (Aktionen) die Option View (Anzeigen) aus.

Auf der Detailseite wird der Grund für den Fehlschlag angezeigt. Wenn die Fehlermeldung „Unable to access object at location...“ oder „Access denied on object at location...“ lautet, überprüfen Sie die folgenden Anforderungen:

- Das Benutzerkonto, mit dem Sie angemeldet sind, muss Leseberechtigungen für das Blueprint-ZIP-Archiv in Amazon S3 haben.
- Der Amazon S3 S3-Bucket, der das ZIP-Archiv enthält, muss über eine Bucket-Richtlinie verfügen, die Ihrer AWS Konto-ID Leseberechtigungen für das Objekt gewährt. Weitere Informationen finden Sie unter Entwicklung von Bauplänen in AWS Glue.
- Der verwendete Amazon-S3-Bucket muss sich in derselben Region befinden, bei der Sie in der Konsole angemeldet sind.

9. Stellen Sie sicher, dass Datenanalysten die nötigen Berechtigungen für den Blueprint haben.

   Die vorgeschlagene IAM-Richtlinie für Datenanalysten finden Sie unter Datenanalystenberechtigungen für Blueprints. Diese Richtlinie gewährt `glue:GetBlueprint` für jede Ressource. Sollte Ihre Richtlinie auf Ressourcenebene differenzierter sein, erteilen Sie Datenanalysten Berechtigungen für diese neu erstellte Ressource.

Um einen Blueprint (AWS CLI) zu registrieren

1. Geben Sie den folgenden Befehl ein.

```
aws glue create-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Geben Sie zur Prüfung des Blueprint-Status den folgenden Befehl ein. Wiederholen Sie den Befehl, bis sich der Status zu `ACTIVE` oder `FAILED` ändert.

```
aws glue get-blueprint --name <blueprint-name>
```

   Wenn der Status `FAILED` lautet und die Fehlermeldung „Unable to access object at location...“ oder „Access denied on object at location...“ angezeigt wird, überprüfen Sie die folgenden Anforderungen:

   - Das Benutzerkonto, mit dem Sie angemeldet sind, muss Leseberechtigungen für das Blueprint-ZIP-Archiv in Amazon S3 haben.

- Der Amazon S3 S3-Bucket, der das ZIP-Archiv enthält, muss über eine Bucket-Richtlinie verfügen, die Ihrer AWS Konto-ID Leseberechtigungen für das Objekt gewährt. Weitere Informationen finden Sie unter Veröffentlichen eines Blueprints.
- Der verwendete Amazon-S3-Bucket muss sich in derselben Region befinden, bei der Sie in der Konsole angemeldet sind.

Weitere Informationen finden Sie auch unter:

- Überblick über Baupläne in AWS Glue

Baupläne anzeigen in AWS Glue

Rufen Sie einen Blueprint auf, um die Beschreibung, den Status und die Parameterspezifikationen zu überprüfen und das Blueprint-ZIP-Archiv herunterzuladen.

Sie können einen Blueprint anzeigen, indem Sie AWS Glue Konsole, AWS Glue API oder AWS Command Line Interface (AWS CLI).

Einen Blueprint aufrufen (Konsole)

1. Öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich die Option Blueprints aus.
3. Wählen Sie auf der Seite Blueprints einen Blueprint aus. Wählen Sie dann im Menü Actions (Aktionen) die Option View (Anzeigen) aus.

So zeigen Sie einen Blueprint an (AWS CLI)

- Geben Sie den folgenden Befehl ein, um nur den Namen, die Beschreibung und den Status des Blueprints anzuzeigen. *<blueprint-name>*Ersetzen Sie ihn durch den Namen des Blueprints, den Sie anzeigen möchten.

```
aws glue get-blueprint --name <blueprint-name>
```

  Die Befehlsausgabe sieht ungefähr wie folgt aus.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "BlueprintLocation": "s3://amzn-s3-demo-bucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
}
```

Geben Sie den folgenden Befehl ein, um auch die Parameterspezifikationen aufzurufen.

```
aws glue get-blueprint --name <blueprint-name>  --include-parameter-spec
```

Die Befehlsausgabe sieht ungefähr wie folgt aus.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "ParameterSpec": "{\"WorkflowName\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null},
\"PassRole\":{\"type\":\"String\",\"collection\":false,\"description\":null,
\"defaultValue\":null,\"allowedValues\":null},\"DynamoDBTableName\":{\"type
\":\"String\",\"collection\":false,\"description\":null,\"defaultValue\":null,
\"allowedValues\":null},\"ScriptLocation\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null}}",
        "BlueprintLocation": "s3://awsexamplebucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
}
```

Fügen Sie das `--include-blueprint` Argument hinzu, um eine URL in die Ausgabe aufzunehmen, die Sie in Ihren Browser einfügen können, um das Blueprint-ZIP-Archiv herunterzuladen AWS Glue gespeichert.

Weitere Informationen finden Sie auch unter:

- Überblick über Baupläne in AWS Glue

Aktualisierung eines Blueprints in AWS Glue

Sie können einen Blueprint aktualisieren, wenn ein überarbeitetes Layoutskript, ein überarbeiteter Satz von Blueprint-Parametern oder überarbeitete zugehörige Dateien vorliegen. Beim Aktualisieren eines Blueprints wird eine neue Version erstellt.

Das Aktualisieren eines Blueprints wirkt sich nicht auf bestehende Workflows aus, die mit dem Blueprint erstellt wurden.

Sie können einen Blueprint aktualisieren, indem Sie AWS Glue Konsole, AWS Glue API oder AWS Command Line Interface (AWS CLI).

Das folgende Verfahren geht davon aus, dass AWS Glue Der Entwickler hat ein aktualisiertes Blueprint-ZIP-Archiv erstellt und auf Amazon S3 hochgeladen.

Einen Blueprint aktualisieren (Konsole)

1. Stellen Sie sicher, dass Sie Leseberechtigungen (`s3:GetObject`) für das Blueprint-ZIP-Archiv in Amazon S3 haben.
2. Öffnen Sie AWS Glue Konsole bei. https://console.aws.amazon.com/glue/

   Melden Sie sich als Benutzer mit Berechtigungen zum Aktualisieren eines Blueprints an. Wechseln Sie zur AWS -Region des Amazon-S3-Buckets, der das Blueprint-ZIP-Archiv enthält.
3. Wählen Sie im Navigationsbereich die Option Blueprints aus.
4. Wählen Sie auf der Seite Blueprints einen Blueprint aus und gehen Sie dann im Menü Actions (Aktionen) auf Edit (Bearbeiten).
5. Ändern Sie auf der Seite Edit a blueprint (Blueprint bearbeiten) die Description (Beschreibung) oder die ZIP archive location (S3) (Speicherort des ZIP-Archivs (S3)). Achten Sie darauf, den Archivdateinamen in den Pfad einzugeben.
6. Wählen Sie Save (Speichern) aus.

Die Seite Blueprints wird wieder geöffnet und zeigt den Blueprint-Status `UPDATING` an. Wählen Sie die Schaltfläche „Refresh“ (Aktualisieren) aus, bis sich der Status in `ACTIVE` oder `FAILED` ändert.

7. Wenn der Status `FAILED` lautet, wählen Sie den Blueprint und dann im Menü Actions (Aktionen) die Option View (Anzeigen) aus.

   Auf der Detailseite wird der Grund für den Fehlschlag angezeigt. Wenn die Fehlermeldung „Unable to access object at location...“ oder „Access denied on object at location...“ lautet, überprüfen Sie die folgenden Anforderungen:

   - Das Benutzerkonto, mit dem Sie angemeldet sind, muss Leseberechtigungen für das Blueprint-ZIP-Archiv in Amazon S3 haben.
   - Der Amazon S3 S3-Bucket, der das ZIP-Archiv enthält, muss über eine Bucket-Richtlinie verfügen, die Ihrer AWS Konto-ID Leseberechtigungen für das Objekt gewährt. Weitere Informationen finden Sie unter Veröffentlichen eines Blueprints.
   - Der verwendete Amazon-S3-Bucket muss sich in derselben Region befinden, bei der Sie in der Konsole angemeldet sind.

> **Note**
>
> Sollte die Aktualisierung fehlschlagen, wird bei der nächsten Blueprint-Ausführung die neueste Version des Blueprints verwendet, die erfolgreich registriert oder aktualisiert wurde.

Einen Blueprint aktualisieren (AWS CLI)

1. Geben Sie den folgenden Befehl ein.

```
aws glue update-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Geben Sie zur Prüfung des Blueprint-Status den folgenden Befehl ein. Wiederholen Sie den Befehl, bis sich der Status zu `ACTIVE` oder `FAILED` ändert.

```
aws glue get-blueprint --name <blueprint-name>
```

Wenn der Status `FAILED` lautet und die Fehlermeldung „Unable to access object at location...“ oder „Access denied on object at location...“ angezeigt wird, überprüfen Sie die folgenden Anforderungen:

- Das Benutzerkonto, mit dem Sie angemeldet sind, muss Leseberechtigungen für das Blueprint-ZIP-Archiv in Amazon S3 haben.
- Der Amazon S3 S3-Bucket, der das ZIP-Archiv enthält, muss über eine Bucket-Richtlinie verfügen, die Ihrer AWS Konto-ID Leseberechtigungen für das Objekt gewährt. Weitere Informationen finden Sie unter Veröffentlichen eines Blueprints.
- Der verwendete Amazon-S3-Bucket muss sich in derselben Region befinden, bei der Sie in der Konsole angemeldet sind.

Weitere Informationen finden Sie auch unter

- Überblick über Baupläne in AWS Glue

Erstellen eines Workflows aus einem Blueprint in AWS Glue

Sie können einen erstellen AWS Glue Sie können einen Workflow manuell hinzufügen, indem Sie jeweils eine Komponente hinzufügen, oder Sie können einen Workflow aus einem AWS Glue Blaupause. AWS Glue enthält Entwürfe für allgemeine Anwendungsfälle. Ihre AWS Glue Entwickler können zusätzliche Blueprints erstellen.

Important

Beschränken Sie die Gesamtzahl der Aufträge, Crawler und Trigger innerhalb eines Workflows auf 100 oder weniger. Wenn Sie mehr als 100 einschließen, werden möglicherweise Fehler angezeigt, wenn Sie versuchen, Workflow-Läufe fortzusetzen oder zu beenden.

Mit einem Blueprint als Ausgangspunkt können Sie schnell einen Workflow für einen bestimmten Anwendungsfall basierend auf dem im Blueprint definierten generalisierten Anwendungsfall generieren. Sie definieren den spezifischen Anwendungsfall, indem Sie Werte für die Blueprint-

Parameter angeben. Ein Blueprint, der einen Datensatz partitioniert, könnte beispielsweise die Amazon-S3-Quell- und Zielpfade als Parameter enthalten.

AWS Glue erstellt einen Workflow aus einem Blueprint, indem der Blueprint ausgeführt wird. Im ausgeführten Blueprint werden die angegebenen Parameterwerte gespeichert und der Fortschritt und das Ergebnis der Erstellung des Workflows und seiner Komponenten festgehalten. Bei der Problembehandlung eines Workflows können Sie die Blueprint-Ausführung anzeigen, um die Parameterwerte zu ermitteln, die zum Erstellen eines Workflows verwendet wurden.

Zum Erstellen und Anzeigen von Workflows benötigen Sie bestimmte IAM-Berechtigungen. Einen Vorschlag für eine IAM-Richtlinie finden Sie unter Datenanalystenberechtigungen für Blueprints.

Sie können einen Workflow aus einem Blueprint erstellen, indem Sie AWS Glue Konsole, AWS Glue API oder AWS Command Line Interface (AWS CLI).

Einen Workflow aus einem Blueprint erstellen (Konsole)

1. Öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.

   Melden Sie sich als Benutzer mit den Berechtigungen zum Erstellen eines Workflows an.
2. Wählen Sie im Navigationsbereich die Option Blueprints aus.
3. Wählen Sie einen Blueprint aus und dann auf der Registerkarte Actions (Aktionen) die Option Create workflow (Workflow erstellen).
4. Geben Sie auf der Seite Create a workflow from <blueprint-name> (Einen Workflow aus <blueprint-name> erstellen) folgende Informationen ein:

   Blueprint-Parameter

   Diese variieren je nach Blueprint-Design. Bei Fragen zu den Parametern wenden Sie sich an den Entwickler. Blueprints enthalten in der Regel einen Parameter für den Workflow-Namen.

   IAM-Rolle

   Die Rolle, die AWS Glue geht davon aus, den Workflow und seine Komponenten zu erstellen. Die Rolle muss über Berechtigungen zum Erstellen und Löschen von Workflows, Aufträgen, Crawlern und Auslösern verfügen. Einen Vorschlag für eine Richtlinie für die Rolle finden Sie unter Berechtigungen für Blueprint-Rollen.
5. Wählen Sie Absenden aus.

Die Seite Blueprint Details (Blueprint-Details) mit einer Liste der Blueprint-Ausführungen wird angezeigt.

6. Überprüfen Sie in der Liste der Blueprint-Ausführungen den obersten Eintrag auf den Status der Workflow-Erstellung.

   Der ursprüngliche Status ist `RUNNING`. Wählen Sie die Schaltfläche „Refresh“ (Aktualisieren) aus, bis sich der Status in `SUCCEEDED` oder `FAILED` ändert.

7. Führen Sie eine der folgenden Aktionen aus:

   - Wenn der Abschlussstatus `SUCCEEDED` ist, können Sie die Seite Workflows öffnen, den neu erstellten Workflow auswählen und ihn ausführen. Bevor Sie den Workflow ausführen, können Sie das Designdiagramm überprüfen.
   - Wenn der Abschlussstatus `FAILED` ist, wählen Sie die Blueprint-Ausführung und dann im Menü Actions (Aktionen) die Option View (Anzeigen) aus, um die Fehlermeldung anzuzeigen.

Weitere Informationen zu Workflows und Blueprints finden Sie in den folgenden Themen.

- Überblick über Workflows in AWS Glue
- Aktualisierung eines Blueprints in AWS Glue
- Manuelles Erstellen und Ausarbeiten eines Workflows in AWS Glue

Das Anzeigen von Blueprint läuft in AWS Glue

Rufen Sie eine Blueprint-Ausführung auf, um die folgenden Informationen anzuzeigen:

- den Namen des erstellten Workflows
- Blueprint-Parameterwerte, die zum Erstellen des Workflows verwendet wurden.
- den Status der Workflow-Erstellung

Sie können einen Blueprint-Lauf anzeigen, indem Sie AWS Glue Konsole, AWS Glue API oder AWS Command Line Interface (AWS CLI).

Eine Blueprint-Ausführung aufrufen (Konsole)

1. Öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.

2. Wählen Sie im Navigationsbereich die Option Blueprints aus.
3. Wählen Sie auf der Seite Blueprints einen Blueprint aus. Wählen Sie dann im Menü Actions (Aktionen) die Option View (Anzeigen) aus.
4. Wählen Sie unten auf der Seite Blueprint Details (Blueprint-Details) eine Blueprint-Ausführung aus und gehen Sie im Menü Actions (Aktionen) auf View (Anzeigen).

Eine Blueprint-Ausführung aufrufen (AWS CLI)

- Geben Sie den folgenden Befehl ein. *<blueprint-name>*Ersetzen Sie es durch den Namen des Blueprints. Ersetzen Sie es *<blueprint-run-id>* durch die Blueprint-Run-ID.

```
aws glue get-blueprint-run --blueprint-name <blueprint-name> --run-id <blueprint-run-id>
```

Weitere Informationen finden Sie auch unter:

- Überblick über Baupläne in AWS Glue

AWS CloudFormation für AWS Glue

AWS CloudFormation ist ein Dienst, der viele AWS Ressourcen erstellen kann. AWS Gluebietet API-Operationen zum Erstellen von Objekten in der AWS Glue Data Catalog. Es kann jedoch praktischer sein, AWS Glue Objekte und andere zugehörige AWS Ressourcenobjekte in einer AWS CloudFormation Vorlagendatei zu definieren und zu erstellen. Anschließend können Sie den Prozess der Objekterstellung automatisieren.

AWS CloudFormation bietet eine vereinfachte Syntax — entweder JSON (JavaScript Object Notation) oder YAML (YAML Ain't Markup Language) —, um die Erstellung von Ressourcen auszudrücken. AWS Sie können mit AWS CloudFormation -Vorlagen Data-Catalog-Objekte wie Datenbanken, Tabellen, Partitionen, Crawlers, Classifier und Verbindungen definieren. Darüber hinaus können Sie ETL-Objekte wie Aufträge, Auslöser und Entwicklungsendpunkte definieren. Sie erstellen eine Vorlage, die alle gewünschten AWS Ressourcen beschreibt und sich um die Bereitstellung und Konfiguration dieser Ressourcen für Sie AWS CloudFormation kümmert.

Weitere Informationen finden Sie unter Was ist AWS CloudFormation? und Arbeiten mit AWS CloudFormation Vorlagen im AWS CloudFormation Benutzerhandbuch.

Wenn Sie beabsichtigen, AWS CloudFormation Vorlagen zu verwendenAWS Glue, die mit kompatibel sind, müssen Sie als Administrator Zugriff AWS CloudFormation auf die AWS Dienste und Aktionen gewähren, von denen dies abhängt. Um Berechtigungen zum Erstellen von AWS CloudFormation Ressourcen zu gewähren, fügen Sie Benutzern, die mit arbeiten, die folgende Richtlinie an AWS CloudFormation:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Die folgende Tabelle enthält die Aktionen, die eine AWS CloudFormation Vorlage in Ihrem Namen ausführen kann. Sie enthält Links zu Informationen zu den AWS Ressourcentypen und ihren Eigenschaftstypen, die Sie einer AWS CloudFormation Vorlage hinzufügen können.

AWS Glue-Ressource	AWS CloudFormation Vorlage	AWS Glue-Beispiele
Classifier	AWS::Glue::Classifier	Grok-Classifier, JSON-Classifier, XML-Classifier
Verbindung	AWS::Glue::Connection	MySQL-Verbindung
Crawler	AWS::Glue::Crawler	Amazon-S3-Crawler, MySQL-Crawler
Datenbank	AWS::Glue::Database	Leere Datenbank, Datenbank mit Tabellen
Entwicklungsendpunkt	AWS::Glue::DevEndpoint	Entwicklungsendpunkt
Aufgabe	AWS::Glue::Job	Amazon-S3-Auftrag, JDBC-Auftrag
Machine-Learning-Transformation	AWS::Glue::MLTransform	Machine-Learning-Transformation
Regelsatz für die Datenqualität	AWS::Glue::DataQualityRegelsatz	Regelsatz für Datenqualität, Regelsatz für Datenqualität mit Scheduler EventBridge
Partition	AWS::Glue::Partition	Tabellenpartitionen
Tabelle	AWS::Glue::Table	Tabelle in einer Datenbank
Auslöser	AWS::Glue::Trigger	Auslöser nach Bedarf Geplanter Auslöser, Bedingter Auslöser

Verwenden Sie für Ihre ersten Schritte die folgenden Beispielvorlagen und passen Sie diese mit Ihren eigenen Metadaten an. Verwenden Sie dann die AWS CloudFormation Konsole, um einen AWS

CloudFormation Stack zum Hinzufügen von Objekten AWS Glue und allen zugehörigen Diensten zu erstellen. Viele Felder in einem AWS Glue-Objekt sind optional. Diese Vorlagen veranschaulichen die Felder, die für ein funktionierendes und funktionales AWS Glue-Objekt erforderlich sind.

Eine AWS CloudFormation Vorlage kann entweder im JSON- oder YAML-Format vorliegen. In den Beispielen wird für eine bessere Lesbarkeit YAML verwendet. Die Beispiele enthalten Kommentare (#) zur Beschreibung der Werte, die in den Vorlagen definiert sind.

AWS CloudFormation Vorlagen können einen `Parameters` Abschnitt enthalten. Dieser Abschnitt kann im Beispieltext oder wenn die YAML-Datei an die AWS CloudFormation Konsole gesendet wird, um einen Stack zu erstellen, geändert werden. Der `Resources` Abschnitt der Vorlage enthält die Definition von AWS Glue und verwandten Objekten. AWS CloudFormation Definitionen der Vorlagensyntax können Eigenschaften enthalten, die eine detailliertere Eigenschaftssyntax enthalten. Möglicherweise sind nicht alle Eigenschaften erforderlich, um ein AWS Glue-Objekt zu erstellen. Diese Beispiele zeigen Beispielwerte für allgemeine Eigenschaften zum Generieren eines AWS Glue-Objekts.

AWS CloudFormation Beispielvorlage für eine AWS Glue Datenbank

Eine AWS Glue-Datenbank im Data Catalog enthält Metadatentabellen. Die Datenbank besteht aus sehr wenigen Eigenschaften und kann im Datenkatalog mit einer AWS CloudFormation Vorlage erstellt werden. Die folgende Beispielvorlage soll Ihnen den Einstieg erleichtern und die Verwendung von AWS CloudFormation Stacks mit AWS Glue veranschaulichen. Die einzige Ressource, die von der Beispielvorlage erstellt wird, ist eine Datenbank namens `cfn-mysampledatabase`. Sie können sie ändern, indem Sie den Text des Beispiels bearbeiten oder den Wert auf der AWS CloudFormation Konsole ändern, wenn Sie die YAML einreichen.

Das folgende Beispiel zeigt Beispielwerte für allgemeine Eigenschaften zum Erstellen einer AWS Glue-Datenbank. Weitere Hinweise zur AWS CloudFormation Datenbankvorlage für finden Sie AWS Glue unter AWS::Glue::Database.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database named
 mysampledatabase
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
```

```
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-mysampledatabse

# Resources section defines metadata for the Data Catalog
Resources:
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      # The database is created in the Data Catalog for your account
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        # The name of the database is defined in the Parameters section above
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
        LocationUri: s3://crawler-public-us-east-1/flight/2016/csv/
        #Parameters: Leave AWS database parameters blank
```

AWS CloudFormation Beispielvorlage für eine AWS Glue Datenbank, Tabelle und Partition

Eine AWS Glue-Tabelle enthält die Metadaten, die die Struktur und den Speicherort der Daten definieren, die Sie mit Ihren ETL-Skripts verarbeiten möchten. Innerhalb einer Tabelle können Sie Partitionen definieren, um das Verarbeiten der Daten parallel auszuführen. Eine Partition ist ein Datenanteil, den Sie mit einem Schlüssel definiert haben. Wenn Sie beispielsweise "Monat" als Schlüssel verwenden, sind alle Daten für Januar in derselben Partition enthalten. In AWS Glue können Datenbanken Tabellen enthalten und Tabellen Partitionen.

Das folgende Beispiel zeigt, wie eine Datenbank, eine Tabelle und Partitionen mithilfe einer AWS CloudFormation -Vorlage gefüllt werden. Das Basis-Datenformat ist das kommaseparierte (,) `csv`-Format. Da eine Datenbank vorhanden sein muss, ehe sie eine Tabelle enthalten kann, und eine Tabelle vorhanden sein muss, ehe Partitionen erstellt werden können, verwendet die Vorlage die `DependsOn`-Anweisung zum Definieren der Abhängigkeit dieser Objekte, wenn sie erstellt werden.

Die Werte in diesem Beispiel definieren eine Tabelle mit Flugdaten aus einem öffentlich verfügbaren Amazon S3 Bucket. Zur Illustrationszwecken sind nur wenige Spalten der Daten und nur ein

Partitionierungsschlüssel definiert. Vier Partitionen sind zudem im Data Catalog definiert. Einige Felder zur Beschreibung der Speicherung der grundlegenden Daten werden ebenfalls in den `StorageDescriptor`-Feldern angezeigt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database, a table,
 and partitions
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
# Parameters substituted in the Resources section
# These parameters are names of the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTableName1:
    Type: String
    Default: cfn-manual-table-flights-1
# Resources to create metadata in the Data Catalog
Resources:
###
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
###
# Create an AWS Glue table
  CFNTableFlights:
    # Creating the table waits for the database to be created
    DependsOn: CFNDatabaseFlights
    Type: AWS::Glue::Table
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableInput:
        Name: !Ref CFNTableName1
        Description: Define the first few columns of the flights table
```

```
        TableType: EXTERNAL_TABLE
        Parameters: {
    "classification": "csv"
  }
#       ViewExpandedText: String
        PartitionKeys:
        # Data is partitioned by month
        - Name: mon
          Type: bigint
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: year
            Type: bigint
          - Name: quarter
            Type: bigint
          - Name: month
            Type: bigint
          - Name: day_of_month
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 1
# Create an AWS Glue partition
  CFNPartitionMon1:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 1
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
```

```
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=1/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 2
# Create an AWS Glue partition
  CFNPartitionMon2:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 2
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=2/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 3
# Create an AWS Glue partition
  CFNPartitionMon3:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 3
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
```

```
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=3/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 4
# Create an AWS Glue partition
  CFNPartitionMon4:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 4
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=4/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Grok-Klassifikator

Ein AWS Glue-Classifier bestimmt das Schema Ihrer Daten. Ein benutzerdefinierter Classifier-Typ verwendet für eine Übereinstimmung mit Ihren Daten ein Grok-Muster. Wenn das Muster übereinstimmt, wird der benutzerdefinierte Classifier für das Erstellen des Tabellenschemas verwendet. Darüber hinaus legt er `classification` auf den in der Classifier-Definition angegebenen Wert fest.

In diesem Beispiel wird ein Classifier generiert, der ein Schema mit einer Spalte namens `message` erstellt und die Klassifizierung auf `greedy` setzt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-grok-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses grok pattern to put all data in one column and classifies
 it as "greedy".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      GrokClassifier:
        #Grok classifier that puts all data in one column
        Name: !Ref CFNClassifierName
        Classification: greedy

        GrokPattern: "%{GREEDYDATA:message}"
        #CustomPatterns: none
```

AWS CloudFormation Beispielvorlage für einen AWS Glue JSON-Klassifikator

Ein AWS Glue-Classifier bestimmt das Schema Ihrer Daten. Ein Typ von benutzerdefiniertem Klassifikator verwendet eine `JsonPath` Zeichenfolge, die die JSON-Daten definiert, die der Klassifikator klassifizieren soll. AWS Glue unterstützt eine Teilmenge der Operatoren für `JsonPath`, wie unter Benutzerdefinierte Klassifikatoren schreiben beschrieben. JsonPath

Wenn das Muster übereinstimmt, wird der benutzerdefinierte Classifier verwendet, um das Schema Ihrer Tabelle zu erstellen.

In diesem Beispiel wird ein Classifier erstellt, der in einem Objekt ein Schema mit jedem Datensatz im `Records3`-Array erstellt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a JSON classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-json-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses a JSON pattern.
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      JSONClassifier:
        #JSON classifier
        Name: !Ref CFNClassifierName
        JsonPath: $.Records3[*]
```

AWS CloudFormation Beispielvorlage für einen AWS Glue XML-Klassifikator

Ein AWS Glue-Classifier bestimmt das Schema Ihrer Daten. Ein Typ von angepasstem Classifier gibt ein XML-Tag an, um das Element festzulegen, das jeden Datensatz in einem XML-Dokument enthält, das analysiert wird. Wenn das Muster übereinstimmt, wird der benutzerdefinierte Classifier für das

Erstellen des Tabellenschemas verwendet. Darüber hinaus legt er `classification` auf den in der Classifier-Definition angegebenen Wert fest.

In diesem Beispiel wird ein Classifier generiert, der ein Schema mit jedem Datensatz in dem `Record`-Tag erstellt und die Klassifizierung auf `XML` setzt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an XML classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-xml-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses the XML pattern and classifies it as "XML".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      XMLClassifier:
        #XML classifier
        Name: !Ref CFNClassifierName
        Classification: XML
        RowTag: <Records>
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Crawler für Amazon S3

Ein AWS Glue-Crawler erstellt Metadatentabellen in Ihrem Data Catalog, die Ihren Daten entsprechen. Anschließend können Sie diese Tabellendefinitionen als Quellen und Ziele in Ihren ETL-Aufträgen verwenden.

Dieses Beispiel erstellt einen Crawler, die erforderliche IAM-Rolle sowie eine AWS Glue-Datenbank im Data Catalog. Wenn dieser Crawler ausgeführt wird, nimmt er die IAM-Rolle an und erstellt eine Tabelle in der Datenbank für öffentliche Flugdaten. Die Tabelle wird mit dem Präfix "`cfn_sample_1_`" generiert. Die von dieser Vorlage erstellte IAM-Rolle lässt globale Berechtigungen zu. Möglicherweise möchten Sie eine benutzerdefinierte Rolle erstellen. Von diesem Classifier werden keine benutzerdefinierten Classifier erstellt. Standardmäßig werden integrierte AWS Glue-Classifier verwendet.

Wenn Sie dieses Beispiel an die AWS CloudFormation Konsole senden, müssen Sie bestätigen, dass Sie die IAM-Rolle erstellen möchten.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-flights-1
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTablePrefixName:
    Type: String
    Default: cfn_sample_1_
#
#
# Resources section defines metadata for the Data Catalog
Resources:
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
```

```
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
# Create a database to contain tables created by the crawler
 CFNDatabaseFlights:
   Type: AWS::Glue::Database
   Properties:
     CatalogId: !Ref AWS::AccountId
     DatabaseInput:
       Name: !Ref CFNDatabaseName
       Description: "AWS Glue container to hold metadata tables for the flights
crawler"
#Create a crawler to crawl the flights data on a public S3 bucket
 CFNCrawlerFlights:
   Type: AWS::Glue::Crawler
   Properties:
     Name: !Ref CFNCrawlerName
     Role: !GetAtt CFNRoleFlights.Arn
     #Classifiers: none, use the default classifier
     Description: AWS Glue crawler to crawl flights data
     #Schedule: none, use default run-on-demand
     DatabaseName: !Ref CFNDatabaseName
     Targets:
       S3Targets:
         # Public S3 bucket with the flights data
         - Path: "s3://crawler-public-us-east-1/flight/2016/csv"
     TablePrefix: !Ref CFNTablePrefixName
     SchemaChangePolicy:
       UpdateBehavior: "UPDATE_IN_DATABASE"
```

```
        DeleteBehavior: "LOG"
      Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

AWS CloudFormation Beispielvorlage für eine Verbindung AWS Glue

Eine AWS Glue-Verbindung im Data Catalog enthält die JDBC- und Netzwerkdaten, die für die Verbindung mit einer JDBC-Datenbank erforderlich sind. Diese Informationen werden verwendet, wenn Sie eine Verbindung mit einer JDBC-Datenbank herstellen, um ETL-Aufträge auszuführen oder zu durchsuchen.

In diesem Beispiel wird eine Verbindung zu einer Amazon-RDS-MySQL-Datenbank namens `devdb` hergestellt. Bei Verwendung dieser Verbindung müssen auch eine IAM-Rolle sowie Datenbank-Anmeldeinformationen und Netzwerkverbindungswerte bereitgestellt werden. Weitere Informationen finden Sie in den Details zu den erforderlichen Feldern in der Vorlage.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a connection
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the connection to be created
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
  CFNJDBCString:
    Type: String
    Default: "jdbc:mysql://xxx-mysql.yyyyyyyyyyyyyy.us-east-1.rds.amazonaws.com:3306/
devdb"
  CFNJDBCUser:
    Type: String
    Default: "master"
  CFNJDBCPassword:
    Type: String
```

```
    Default: "12345678"
    NoEcho: true
#
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNConnectionMySQL:
    Type: AWS::Glue::Connection
    Properties:
      CatalogId: !Ref AWS::AccountId
      ConnectionInput:
        Description: "Connect to MySQL database."
        ConnectionType: "JDBC"
        #MatchCriteria: none
        PhysicalConnectionRequirements:
          AvailabilityZone: "us-east-1d"
          SecurityGroupIdList:
           - "sg-7d52b812"
          SubnetId: "subnet-84f326ee"
        ConnectionProperties: {
          "JDBC_CONNECTION_URL": !Ref CFNJDBCString,
          "USERNAME": !Ref CFNJDBCUser,
          "PASSWORD": !Ref CFNJDBCPassword
        }
        Name: !Ref CFNConnectionName
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Crawler für JDBC

Ein AWS Glue-Crawler erstellt Metadatentabellen in Ihrem Data Catalog, die Ihren Daten entsprechen. Anschließend können Sie diese Tabellendefinitionen als Quellen und Ziele in Ihren ETL-Aufträgen verwenden.

Dieses Beispiel erstellt einen Crawler, die erforderliche IAM-Rolle sowie eine AWS Glue-Datenbank im Data Catalog. Wenn dieser Crawler ausgeführt wird, nimmt er die IAM-Rolle an und erstellt eine Tabelle in der Datenbank für öffentliche Flugdaten, die in einer MySQL-Datenbank gespeichert wurden. Die Tabelle wird mit dem Präfix "`cfn_jdbc_1_`" generiert. Die von dieser Vorlage erstellte IAM-Rolle lässt globale Berechtigungen zu. Möglicherweise möchten Sie eine benutzerdefinierte Rolle erstellen. Für JDBC-Daten können keine benutzerdefinierten Classifier definiert werden. Standardmäßig werden integrierte AWS Glue-Classifier verwendet.

Wenn Sie dieses Beispiel an die AWS CloudFormation Konsole senden, müssen Sie bestätigen, dass Sie die IAM-Rolle erstellen möchten.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-jdbc-flights-1
# The name of the database to be created to contain tables
  CFNDatabaseName:
    Type: String
    Default: cfn-database-jdbc-flights-1
# The prefix for all tables crawled and created
  CFNTablePrefixName:
    Type: String
    Default: cfn_jdbc_1_
# The name of the existing connection to the MySQL database
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
# The name of the JDBC path (database/schema/table) with wildcard (%) to crawl
  CFNJDBCPath:
    Type: String
    Default: saldev/%
#
#
# Resources section defines metadata for the Data Catalog
Resources:
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
```

```
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
# Create a database to contain tables created by the crawler
 CFNDatabaseFlights:
   Type: AWS::Glue::Database
   Properties:
     CatalogId: !Ref AWS::AccountId
     DatabaseInput:
       Name: !Ref CFNDatabaseName
       Description: "AWS Glue container to hold metadata tables for the flights
crawler"
#Create a crawler to crawl the flights data in MySQL database
 CFNCrawlerFlights:
   Type: AWS::Glue::Crawler
   Properties:
     Name: !Ref CFNCrawlerName
     Role: !GetAtt CFNRoleFlights.Arn
     #Classifiers: none, use the default classifier
     Description: AWS Glue crawler to crawl flights data
     #Schedule: none, use default run-on-demand
     DatabaseName: !Ref CFNDatabaseName
     Targets:
       JdbcTargets:
         # JDBC MySQL database with the flights data
         - ConnectionName: !Ref CFNConnectionName
           Path: !Ref CFNJDBCPath
         #Exclusions: none
```

```
      TablePrefix: !Ref CFNTablePrefixName
      SchemaChangePolicy:
        UpdateBehavior: "UPDATE_IN_DATABASE"
        DeleteBehavior: "LOG"
   Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Job für Amazon S3 auf Amazon S3

Ein AWS Glue-Auftrag im Data Catalog enthält die Parameterwerte, die zum Ausführen eines Skripts in AWS Glue benötigt werden.

Dieses Beispiel erstellt einen Auftrag, der Flugdaten aus einem Amazon S3 Bucket im `csv`-Format ausliest und diese in eine Amazon-S3-Parquet-Datei schreibt. Das Skript, das von diesen Auftrag ausgeführt wird, muss bereits vorhanden sein. Sie können mit der AWS Glue-Konsole ein ETL-Skript für Ihre Umgebung erstellen. Wenn dieser Auftrag ausgeführt wird, muss auch eine IAM-Rolle mit den richtigen Berechtigungen bereitgestellt werden.

Häufige Parameterwerte werden in der Vorlage gezeigt. Zum Beispiel ist `AllocatedCapacity` (DPUs) standardmäßig auf 5 voreingestellt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using the public flights S3 table in a
 public bucket
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-2
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
```

```
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-test2
#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create job to run script which accesses flightscsv table and write to S3 file as
 parquet.
# The script already exists and is called by this job
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
      # If script written in Scala, then set DefaultArguments={'--job-language';
 'scala', '--class': 'your scala class'}
      #Connections:  No connection needed for S3 to S3 job
      #  ConnectionsList
      #MaxRetries: Double
      Description: Job created with CloudFormation
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
 and folders that begin with "aws-glue-"
             # script uses temp directory from job definition if required (temp
 directory not used S3 to S3)
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Job für JDBC zu Amazon S3

Ein AWS Glue-Auftrag im Data Catalog enthält die Parameterwerte, die zum Ausführen eines Skripts in AWS Glue benötigt werden.

In diesem Beispiel wird ein Auftrag für das Lesen von Flugdaten aus einer MySQL-JDBC-Datenbank erstellt, wie durch die Verbindung namens `cfn-connection-mysql-flights-1` definiert, und in eine Amazon-S3-Parquet-Datei geschrieben. Das Skript, das von diesen Auftrag ausgeführt wird, muss bereits vorhanden sein. Sie können mit der AWS Glue-Konsole ein ETL-Skript für Ihre Umgebung erstellen. Wenn dieser Auftrag ausgeführt wird, muss auch eine IAM-Rolle mit den richtigen Berechtigungen bereitgestellt werden.

Häufige Parameterwerte werden in der Vorlage gezeigt. Zum Beispiel ist `AllocatedCapacity` (DPUs) standardmäßig auf 5 voreingestellt.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using a MySQL JDBC DB with the flights
 data to an S3 file
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-JDBC-to-S3-1
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-dec4a
# The name of the connection used for JDBC data source
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create job to run script which accesses JDBC flights table via a connection and write
 to S3 file as parquet.
```

```
# The script already exists and is called by this job
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
      # For example, if required by script, set temporary directory as
 DefaultArguments={'--TempDir'; 's3://aws-glue-temporary-xyc/sal'}
      Connections:
        Connections:
        - !Ref CFNConnectionName
      #MaxRetries: Double
      Description: Job created with CloudFormation using existing script
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
 and folders that begin with "aws-glue-"
             # if required, script defines temp directory as argument TempDir and used
 in script like redshift_tmp_dir = args["TempDir"]
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

AWS CloudFormation Beispielvorlage für einen AWS Glue On-Demand-Trigger

Ein AWS Glue-Auslöser im Data Catalog enthält die Parameterwerte, die zum Starten einer Auftragsausführung erforderlich sind, wenn der Auslöser ausgelöst wird. Ein Auslöser nach Bedarf wird ausgelöst, wenn Sie ihn aktivieren.

In diesem Beispiel wird ein Auslöser nach Bedarf erstellt, der einen Auftrag namens `cfn-job-S3-to-S3-1` startet.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an on-demand trigger
```

```
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-ondemand-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating an on-demand trigger for a job
Resources:
# Create trigger to run an existing job (CFNJobName) on an on-demand schedule.
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: ON_DEMAND
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule:
      #Predicate:
```

AWS CloudFormation Beispielvorlage für einen AWS Glue geplanten Trigger

Ein AWS Glue-Auslöser im Data Catalog enthält die Parameterwerte, die zum Starten einer Auftragsausführung erforderlich sind, wenn der Auslöser ausgelöst wird. Ein geplanter Auslöser wird ausgelöst, wenn er aktiviert und der Cron-Timer angezeigt wird.

In diesem Beispiel wird ein geplanter Auslöser erstellt, der einen Auftrag namens `cfn-job-S3-to-S3-1` startet. Der Timer ist ein Cron-Ausdruck, durch den der Auftrag an Wochentagen alle 10 Minuten ausgeführt wird.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a scheduled trigger
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-scheduled-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating a scheduled trigger for a job
#
Resources:
# Create trigger to run an existing job (CFNJobName) on a cron schedule.
  TriggerSample1CFN:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: SCHEDULED
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      # # Run the trigger every 10 minutes on Monday to Friday
      Schedule: cron(0/10 * ? * MON-FRI *)
      #Predicate:
```

AWS CloudFormation Beispielvorlage für einen AWS Glue bedingten Trigger

Ein AWS Glue-Auslöser im Data Catalog enthält die Parameterwerte, die zum Starten einer Auftragsausführung erforderlich sind, wenn der Auslöser ausgelöst wird. Ein bedingter Auslöser wird

ausgelöst, wenn er aktiviert und die Bedingungen, beispielsweise ein erfolgreicher Auftragsabschluss, erfüllt sind.

In diesem Beispiel wird ein bedingter Auslöser erstellt, der einen Auftrag namens `cfn-job-S3-to-S3-1` startet. Dieser Auftrag startet, wenn der Auftrag namens `cfn-job-S3-to-S3-2` erfolgreich abgeschlossen ist.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a conditional trigger for a job, which starts
 when another job completes
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The existing job that when it finishes causes trigger to fire
  CFNJobName2:
    Type: String
    Default: cfn-job-S3-to-S3-2
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-conditional-1
#
Resources:
# Create trigger to run an existing job (CFNJobName) when another job completes
 (CFNJobName2).
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: CONDITIONAL
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule: none
```

```
      Predicate:
        #Value for Logical is required if more than 1 job listed in Conditions
        Logical: AND
        Conditions:
          - LogicalOperator: EQUALS
            JobName: !Ref CFNJobName2
            State: SUCCEEDED
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Entwicklungsendpunkt

Eine AWS Glue-Machine-Learning-Transformation ist eine benutzerdefinierte Transformation zur Bereinigung Ihrer Daten. Derzeit ist eine Transformation mit dem Namen verfügbar FindMatches. Die FindMatches Transformation ermöglicht es Ihnen, doppelte oder übereinstimmende Datensätze in Ihrem Datensatz zu identifizieren, selbst wenn die Datensätze keinen gemeinsamen eindeutigen Bezeichner haben und keine Felder exakt übereinstimmen.

Dieses Beispiel erstellt eine Machine-Learning-Transformation. Weitere Hinweise zu den Parametern, die Sie zum Erstellen einer Machine-Learning-Transformation benötigen, finden Sie unter Datensatzabgleich mit AWS Lake Formation FindMatches.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a machine learning transform
#
# Resources section defines metadata for the machine learning transform
Resources:
  MyMLTransform:
    Type: "AWS::Glue::MLTransform"
    Condition: "isGlueMLGARegion"
    Properties:
      Name: !Sub "MyTransform"
      Description: "The bestest transform ever"
      Role: !ImportValue MyMLTransformUserRole
      GlueVersion: "1.0"
      WorkerType: "Standard"
      NumberOfWorkers: 5
      Timeout: 120
      MaxRetries: 1
      InputRecordTables:
```

```
        GlueTables:
          - DatabaseName: !ImportValue MyMLTransformDatabase
            TableName: !ImportValue MyMLTransformTable
      TransformParameters:
        TransformType: "FIND_MATCHES"
        FindMatchesParameters:
          PrimaryKeyColumnName: "testcolumn"
          PrecisionRecallTradeoff: 0.5
          AccuracyCostTradeoff: 0.5
          EnforceProvidedLabels: True
      Tags:
        key1: "value1"
        key2: "value2"
      TransformEncryption:
        TaskRunSecurityConfigurationName: !ImportValue
 MyMLTransformSecurityConfiguration
        MLUserDataEncryption:
          MLUserDataEncryptionMode: "SSE-KMS"
          KmsKeyId: !ImportValue MyMLTransformEncryptionKey
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Data Quality Regelsatz

Ein Regelsatz für die AWS Glue Datenqualität enthält Regeln, die anhand einer Tabelle im Datenkatalog ausgewertet werden können. Sobald der Regelsatz auf Ihrer Zieltabelle platziert ist, können Sie den Data Catalog aufrufen und eine Auswertung ausführen, die Ihre Daten mit den Regeln des Regelsatzes vergleicht. Diese Regeln können von der Auswertung der Zeilenanzahl bis zur Auswertung der referenziellen Integrität Ihrer Daten reichen.

Das folgende Beispiel ist eine CloudFormation Vorlage, die einen Regelsatz mit einer Vielzahl von Regeln für die angegebene Zieltabelle erstellt.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
```

```
    Type: String
    Default: "CFNRulesetName"
  RulesetDescription:
    Type: String
    Default: "CFN DataQualityRuleset"
  # Rules that will be associated with this ruleset
  Rules:
    Type: String
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # Name of database and table within Data Catalog which the ruleset will
  # be applied too
  DatabaseName:
    Type: String
    Default: "ExampleDatabaseName"
  TableName:
    Type: String
    Default: "ExampleTableName"

# Resources section defines metadata for the Data Catalog
Resources:
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: !Ref RulesetDescription
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: !Ref DatabaseName
        TableName: !Ref TableName
```

AWS CloudFormation Beispielvorlage für einen AWS Glue Data Quality Regelsatz mit Scheduler EventBridge

Ein Regelsatz für die AWS Glue Datenqualität enthält Regeln, die anhand einer Tabelle im Datenkatalog ausgewertet werden können. Sobald der Regelsatz auf Ihrer Zieltabelle platziert ist, können Sie den Data Catalog aufrufen und eine Auswertung ausführen, die Ihre Daten mit den Regeln des Regelsatzes vergleicht. Anstatt manuell in den Datenkatalog gehen zu müssen, um den Regelsatz auszuwerten, können Sie unserer CloudFormation Vorlage auch einen EventBridge Scheduler hinzufügen, um diese Regelsatzauswertungen für Sie in einem bestimmten Zeitintervall zu planen.

Das folgende Beispiel ist eine CloudFormation Vorlage, die einen Regelsatz für die Datenqualität und einen EventBridge Scheduler erstellt, um den oben genannten Regelsatz alle fünf Minuten auszuwerten.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
    Type: String
    Default: "CFNRulesetName"
  # Rules that will be associated with this Ruleset
  Rules:
    Type: String
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # The name of the Schedule to be created
  ScheduleName:
    Type: String
    Default: "ScheduleDQRulsetEvaluation"
  # This expression determines the rate at which the Schedule will evaluate
  # your data using the above ruleset
  ScheduleRate:
```

```
    Type: String
    Default: "rate(5 minutes)"
  # The Request that being sent must match the details of the Data Quality Ruleset
  ScheduleRequest:
    Type: String
    Default: '
        { "DataSource": { "GlueTable": { "DatabaseName": "ExampleDatabaseName",
         "TableName": "ExampleTableName" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ ""CFNRulesetName"" ] }
        '

# Resources section defines metadata for the Data Catalog
Resources:
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: "CFN DataQualityRuleset"
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: "ExampleDatabaseName"
        TableName: "ExampleTableName"
  # Create a Scheduler to schedule evaluation runs on the above ruleset
  ScheduleDQEval:
    Type: AWS::Scheduler::Schedule
    Properties:
      Name: !Ref ScheduleName
      Description: "Schedule DataQualityRuleset Evaluations"
      FlexibleTimeWindow:
        Mode: "OFF"
      ScheduleExpression: !Ref ScheduleRate
      ScheduleExpressionTimezone: "America/New_York"
      State: "ENABLED"
      Target:
        # The ARN is the API that will be run, since we want to evaluate our ruleset
        # we want this specific ARN
        Arn: "arn:aws:scheduler:::aws-sdk:glue:startDataQualityRulesetEvaluationRun"
        # Your RoleArn must have approval to schedule
```

```
        RoleArn: "arn:aws:iam::123456789012:role/AWSGlueServiceRoleDefault"
        # This is the Request that is being sent to the Arn
        Input: '
        { "DataSource": { "GlueTable": { "DatabaseName": "sampledb", "TableName":
 "meteorite" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ "TestCFN" ] }
        '
```

AWS CloudFormation Beispielvorlage für einen Entwicklungsendpunkt AWS Glue

Ein AWS Glue-Entwicklungsendpunkt ist eine Umgebung, in der Sie Ihre AWS Glue-Skripts entwickeln und testen können.

Dieses Beispiel erstellt einen Entwicklungsendpunkt mit Netzwerkparameterwerten, die mindestens für eine erfolgreiche Erstellung erforderlich sind. Weitere Informationen zu den Parametern, die Sie zum Einrichten eines Entwicklungsendpunkts benötigen, finden Sie unter Einrichtung von Netzwerken für die Entwicklung für AWS Glue.

Zum Erstellen des Entwicklungsendpunkts stellen Sie einen vorhandenen IAM-Rollen-ARN (Amazon-Ressourcenname) bereit. Geben Sie einen gültigen öffentlichen RSA-Schlüssel an halten Sie den entsprechenden privaten Schlüssel bereit, wenn Sie einen Notebook-Server auf dem Entwicklungsendpunkt erstellen möchten.

> **Note**
>
> Sie verwalten ihn für jeden von Ihnen erstellten Notebook-Server, der mit einem Entwicklungsendpunkt verknüpft ist. Wenn Sie den Entwicklungsendpunkt löschen, müssen Sie daher den AWS CloudFormation Stack auf der AWS CloudFormation Konsole löschen, um den Notebook-Server zu löschen.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a development endpoint
#
# Parameters section contains names that are substituted in the Resources section
```

```
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNEndpointName:
    Type: String
    Default: cfn-devendpoint-1
  CFNIAMRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012/role/AWSGlueServiceRoleGA
#
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNDevEndpoint:
    Type: AWS::Glue::DevEndpoint
    Properties:
      EndpointName: !Ref CFNEndpointName
      #ExtraJarsS3Path: String
      #ExtraPythonLibsS3Path: String
      NumberOfNodes: 5
      PublicKey: ssh-rsa public.....key myuserid-key
      RoleArn: !Ref CFNIAMRoleArn
      SecurityGroupIds:
        - sg-64986c0b
      SubnetId: subnet-c67cccac
```

AWS Glue Leitfaden zur Programmierung

Ein Skript enthält den Code, der Daten aus Quellen extrahiert, transformiert und in Ziele lädt. AWS Glue führt ein Skript aus, wenn es einen Job startet.

AWS Glue ETL-Skripte sind in Python oder Scala codiert. Während alle Auftragstypen in Python geschrieben werden können, können AWS Glue -für-Spark-Aufträge auch in Scala geschrieben werden. Wenn Sie die Quellcodelogik für Ihren Job in automatisch generieren AWS Glue Studio, wird ein Skript erstellt. Sie können dieses Skript bearbeiten oder Ihr eigenes Skript zur Verarbeitung Ihrer ETL-Vorgänge bereitstellen.

Bereitstellen eigener, benutzerdefinierter Skripts

Skripten führen die Arbeit beim Extrahieren, Transformieren und Laden (ETL) durch AWS Glue. Ein Skript wird erstellt, wenn Sie automatisch die Quellcodelogik für einen Auftrag generieren. Sie können dieses Skript entweder bearbeiten oder Ihr eigenes Skript bereitstellen.

Gehen Sie wie folgt vor AWS Glue, um Ihr eigenes benutzerdefiniertes Skript bereitzustellen:

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie die Registerkarte ETL-Aufträge, und zeigen Sie dann den Abschnitt Auftrag erstellen an. Wählen Sie eine Option für den Skript-Editor.
3. Wählen Sie unter This job runs (Dieser Auftrag wird ausgeführt) eine der folgenden Optionen:
   - Einen neuen Skripts mit Boilerplate-Code erstellen
   - Einen vorhandenen Skript hochladen und bearbeiten
4. Wählen Sie auf der Seite Auftragsdetails die IAM-Rolle, die für die Ausführung Ihres benutzerdefinierten Skripts erforderlich ist. Weitere Informationen finden Sie unter Identitäts- und Zugriffsmanagement für AWS Glue.
5. Wählen Sie alle Verbindungen, auf die Ihr Skript verweist. Diese Objekte werden benötigt, um eine Verbindung zu den erforderlichen JDBC-Datenspeichern herzustellen.

   Eine Elastic Network-Schnittstelle ist eine virtuelle Netzwerkschnittstelle, die Sie an eine Instance in einer Virtual Private Cloud (VPC) anhängen können. Wählen Sie die Elastic Network-Schnittstelle, die erforderlich ist, um eine Verbindung mit dem Datenspeicher herzustellen, der im Skript verwendet wird.

6. Stellen Sie zusätzliche Konfigurationen, einschließlich Parameter, bereit, die für Ihren Auftragstyp spezifisch sind. Weitere Informationen zur Konfiguration für Ihren Auftragstyp finden Sie im Abschnitt Erstellung visueller ETL-Jobs.
7. Fügen Sie auf der Registerkarte Skript Ihr benutzerdefiniertes Skript ein oder schreiben Sie es.

Verwenden Sie den Inhalt dieses Abschnitts als Leitfaden für das Schreiben Ihres benutzerdefinierten Skripts.

Weitere Informationen zum Hinzufügen von Jobs finden Sie unterErstellung visueller ETL-Jobs. AWS Glue

step-by-stepAnleitungen finden Sie im Tutorial „Job hinzufügen“ in der AWS Glue Konsole.

Programmieren von Spark-Skripte

AWS Glue macht es einfach, ETL-Skripts (Extrahieren, Transformieren und Laden) zu schreiben oder automatisch zu generieren und sie zusätzlich zu testen und auszuführen. In diesem Abschnitt werden die durch AWS Glue eingeführten Erweiterungen für Apache Spark beschrieben und Beispiele für das Codieren und Ausführen von ETL-Skripts in Python und Scala bereitgestellt.

> **Important**
>
> Verschiedene Versionen von AWS Glue unterstützen verschiedene Versionen von Apache Spark. Ihr benutzerdefiniertes Skript muss mit der unterstützten Apache-Spark-Version kompatibel sein. Informationen zu den Versionen von AWS Glue finden Sie unterGlue version job property.

Themen

- Tutorial: Ein AWS Glue for Spark-Skript schreiben
- Programmieren AWS Sie Glue ETL-Skripte in PySpark
- Programmieren AWS Glue ETL-Skripte in Scala
- Funktionen und Optimierungen für die Programmierung AWS Glue für Spark-ETL-Skripte

Tutorial: Ein AWS Glue for Spark-Skript schreiben

Dieses Tutorial führt Sie in den Prozess des Schreibens von AWS Glue-Skripten ein. Sie können Skripte nach einem Zeitplan mit Aufträgen oder interaktiv mit interaktiven Sitzungen ausführen. Weitere Informationen über Aufträge finden Sie unter Erstellung visueller ETL-Jobs. Weitere Informationen zu interaktiven Sitzungen finden Sie unter the section called “Überblick über AWS Glue interaktive Sessions”.

Der visuelle Editor von AWS Glue Studio bietet eine grafische Oberfläche ohne Code für die Erstellung von AWS Glue-Jobs. AWS Glue Sie Skripte auf visuelle Jobs zurück. Sie bieten Ihnen Zugriff auf die erweiterten Tools, die für die Arbeit mit Apache-Spark-Programmen zur Verfügung stehen. Sie können von einem Glue-Skript aus auf native Spark APIs - und AWS Glue-Bibliotheken zugreifen, die Workflows zum Extrahieren, Transformieren und Laden (ETL) erleichtern. AWS

In diesem Tutorial extrahieren, transformieren und laden Sie einen Datensatz mit Parktickets. Das Skript, das diese Arbeit erledigt, ist in Form und Funktion identisch mit dem Skript, das in Making ETL easier with AWS Glue Studio auf dem AWS Big Data Blog generiert wurde, in dem der visuelle Editor von AWS Glue Studio vorgestellt wird. Wenn Sie dieses Skript in einem Job ausführen, können Sie es mit visuellen Jobs vergleichen und sehen, wie AWS Glue-ETL-Skripts funktionieren. Dies bereitet Sie darauf vor, zusätzliche Funktionen zu verwenden, die in visuellen Aufträgen noch nicht verfügbar sind.

In diesem Tutorial verwenden Sie die Sprache Python und die Bibliotheken. Eine ähnliche Funktionalität ist in Scala verfügbar. Nachdem Sie dieses Tutorial durchgearbeitet haben, sollten Sie in der Lage sein, ein Scala-Beispielskript zu generieren und zu überprüfen, um zu verstehen, wie der Scala AWS Glue-ETL-Skriptschreibprozess durchgeführt wird.

Voraussetzungen

Für dieses Tutorial müssen die folgenden Voraussetzungen erfüllt sein:

- Dieselben Voraussetzungen wie im AWS Glue Studio-Blogbeitrag, in dem Sie aufgefordert werden, eine AWS CloudFormation Vorlage auszuführen.

  Diese Vorlage verwendet den AWS Glue-Datenkatalog, um den in verfügbaren Parkscheindatensatz zu verwaltens`3://aws-bigdata-blog/artifacts/gluestudio/`. Sie erstellt die folgenden Ressourcen, auf die verwiesen wird:

- AWS Glue StudioRolle — auszuführende IAM-Rolle AWS Gluejobs

- AWS Glue StudioAmazon S3Bucket — Name des Amazon S3 S3-Buckets zum Speichern von Blog-bezogenen Dateien
- AWS Glue StudioTickets/YZDB — AWS Glue Datenkatalog-Datenbank
- AWS Glue StudioTableTickets— Datenkatalogtabelle, die als Quelle verwendet werden soll
- AWS Glue StudioTableTrials— Datenkatalogtabelle, die als Quelle verwendet werden soll
- AWS Glue StudioParkingTicketCount — Datenkatalogtabelle, die als Ziel verwendet werden soll
- Das im AWS Glue Studio-Blogbeitrag generierte Skript. Falls sich der Blogbeitrag ändert, finden Sie das Skript auch im folgenden Text.

Generieren eines Beispielskripts

Sie können den Visual Editor von AWS Glue Studio als leistungsstarkes Tool zur Codegenerierung verwenden, um ein Gerüst für das Skript zu erstellen, das Sie schreiben möchten. Mit diesem Tool erstellen Sie ein Beispielskript.

Wenn Sie diese Schritte überspringen möchten, wird das Skript bereitgestellt.

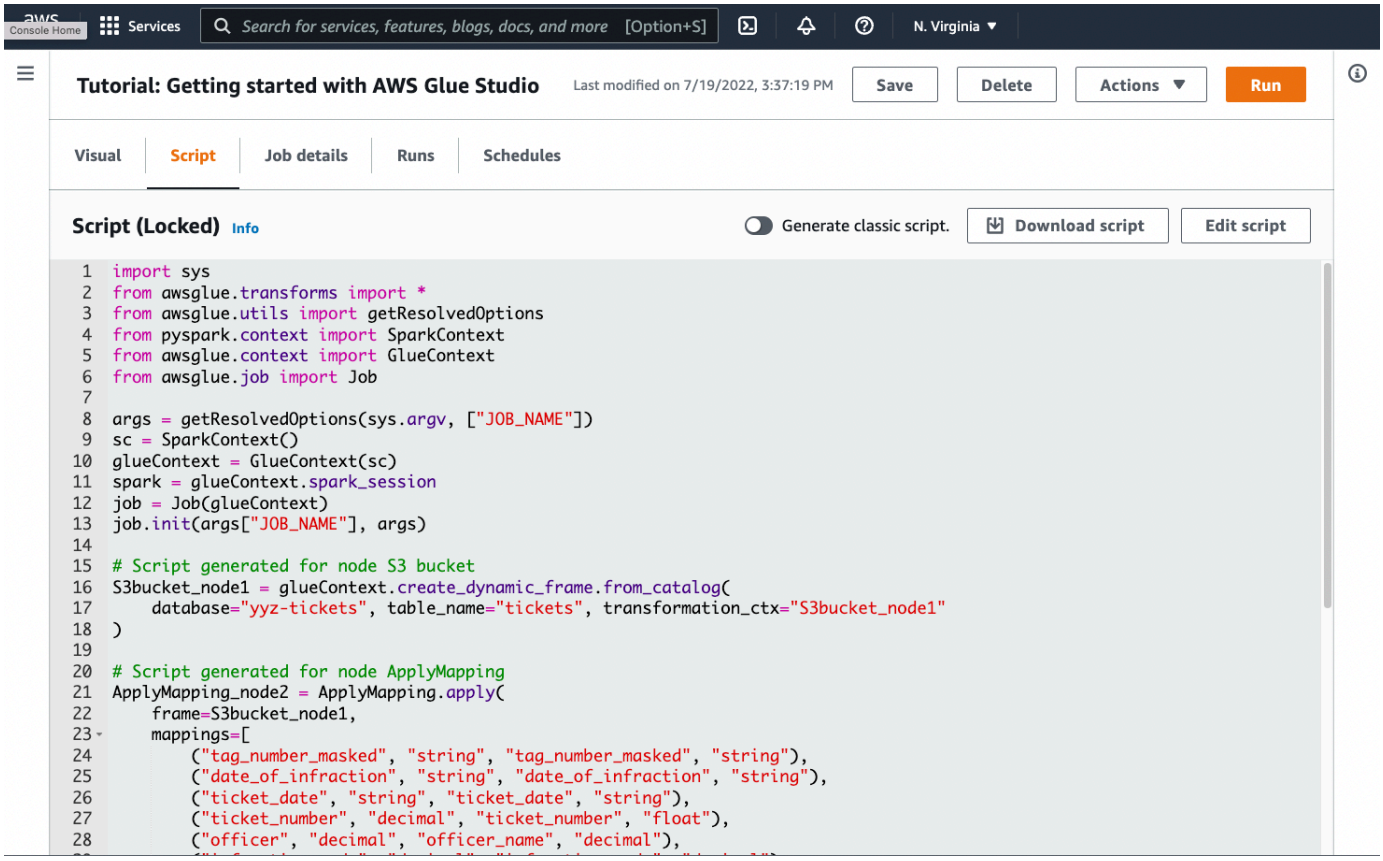
Tutorial-Beispielskript

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 bucket
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
)

# Script generated for node ApplyMapping
ApplyMapping_node2 = ApplyMapping.apply(
```

```
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node S3 bucket
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://DOC-EXAMPLE-BUCKET", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
)

job.commit()
```

So generieren Sie ein Beispielskript

1. Schließe das AWS Glue Studio-Tutorial ab. Um dieses Tutorial abzuschließen, siehe Einen Job in AWS Glue Studio anhand eines Beispieljobs erstellen.
2. Navigieren Sie auf der Auftragsseite zur Registerkarte Script (Skript), wie im folgenden Screenshot gezeigt:



3. Kopieren Sie den vollständigen Inhalt der Registerkarte Script (Skript). Durch Einstellen der Skriptsprache in Job details (Auftragsdetails) können Sie zwischen der Generierung von Python- oder Scala-Code hin und her wechseln.

Schritt 1. Erstellen eines Auftrags und Einfügen Ihres Skripts

In diesem Schritt erstellen Sie einen AWS Glue-Job in der AWS Management Console. Dadurch wird eine Konfiguration eingerichtet, die es AWS Glue ermöglicht, Ihr Skript auszuführen. Es schafft auch einen Ort, an dem Sie Ihr Skript speichern und bearbeiten können.

So erstellen Sie einen --Auftrag

1. Navigieren Sie in der AWS Management Console zur AWS Glue-Landingpage.
2. Wählen Sie im seitlichen Navigationsbereich Jobs (Aufträge) aus.
3. Wählen Sie Spark script editor (Spark-Skript-Editor) in Create job (Auftrag erstellen) und wählen Sie dann Create (Erstellen) aus.
4. Optional – Fügen Sie den vollständigen Text Ihres Skripts in den Bereich Script (Skript) ein. Alternativ können Sie dem Tutorial folgen.

Schritt 2. AWS Glue-Bibliotheken importieren

Sie müssen Ihr Skript so einrichten, dass es mit Code und Konfiguration interagiert, die außerhalb des Skripts definiert sind. Diese Arbeit wird hinter den Kulissen von AWS Glue Studio erledigt.

In diesem Schritt führen Sie die folgenden Aktionen durch.

- Importieren und initialisieren Sie ein `GlueContext`-Objekt. Dies ist beim Schreiben von Skripten der wichtigste Import. Dadurch werden Standardmethoden zum Definieren von Quell- und Zieldatensätzen verfügbar, die der Ausgangspunkt für jedes ETL-Skript sind. Weitere Informationen zur `GlueContext`-Klasse finden Sie unter GlueContext Klasse.
- Initialisieren Sie `SparkContext` und `SparkSession`. Diese ermöglichen es Ihnen, die Spark-Engine zu konfigurieren, die im AWS Glue-Job verfügbar ist. Sie müssen sie nicht direkt in einführenden AWS Glue-Skripten verwenden.
- Rufen Sie `getResolvedOptions` auf, um Ihre Auftragsargumente für die Verwendung innerhalb des Skripts vorzubereiten. Weitere Informationen zum Auflösen von Auftragsparametern finden Sie unter the section called “getResolvedOptions”.
- Initialisieren eines `Job`. Das `Job` Objekt legt die Konfiguration fest und verfolgt den Status verschiedener optionaler AWS Glue-Funktionen. Ihr Skript kann ohne ein `Job`-Objekt ausgeführt werden, aber die beste Vorgehensweise ist, es zu initialisieren. So kommt es später nicht zu Verwirrungen, wenn diese Features später integriert werden.

  Eine dieser Features sind Auftragslesezeichen, die Sie in diesem Tutorial optional konfigurieren können. Im folgenden Abschnitt the section called “Optional – Auftragslesezeichen aktivieren” erfahren Sie mehr über Auftragslesezeichen.

In diesem Verfahren schreiben Sie den folgenden Code. Dieser Code ist ein Teil des generierten Beispielskripts.

```
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
```

```
job = Job(glueContext)
job.init(args["JOB_NAME"], args)
```

Um AWS Glue-Bibliotheken zu importieren

- Kopieren Sie diesen Codeabschnitt und fügen Sie ihn in den Script (Skript)-Editor ein.

> **Note**
>
> Sie sollten bedenken, dass das Kopieren von Code eine schlechte technische Praxis ist. In diesem Tutorial empfehlen wir dies, um Sie zu ermutigen, Ihre Kernvariablen in allen AWS Glue-ETL-Skripten einheitlich zu benennen.

Schritt 3. Extrahieren von Daten aus einer Quelle

Bei jedem ETL-Prozess müssen Sie zunächst einen Quelldatensatz definieren, den Sie ändern möchten. Im visuellen Editor von AWS Glue Studio stellen Sie diese Informationen bereit, indem Sie einen Quellknoten erstellen.

In diesem Schritt geben Sie für die `create_dynamic_frame.from_catalog`-Methode eine `database` und einen `table_name` an, um Daten aus einer Quelle zu extrahieren, die im AWS -Glue-Datenkatalog konfiguriert wurden.

Im vorherigen Schritt haben Sie ein `GlueContext`-Objekt initialisiert. Sie verwenden dieses Objekt, um Methoden zu finden, die zum Konfigurieren von Quellen verwendet werden, z. B `create_dynamic_frame.from_catalog`.

In diesem Verfahren schreiben Sie den folgenden Code mithilfe von `create_dynamic_frame.from_catalog`. Dieser Code ist ein Teil des generierten Beispielskripts.

```
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
    )
```

So extrahieren Sie Daten aus einer Quelle

1. Suchen Sie in der Dokumentation nach einer Methode `GlueContext` zum Extrahieren von Daten aus einer im AWS Glue-Datenkatalog definierten Quelle. Diese Methoden

sind dokumentiert in the section called “GlueContext”. Wählen Sie die Methode create_dynamic_frame.from_catalog. Rufen Sie diese Methode auf `glueContext` auf.

2. Untersuchen Sie die Dokumentation auf `create_dynamic_frame.from_catalog`. Diese Methode erfordert die Parameter `database` und `table_name`. Stellen Sie die notwendigen Parameter für `create_dynamic_frame.from_catalog` bereit.

   Der AWS Glue-Datenkatalog speichert Informationen über den Speicherort und das Format Ihrer Quelldaten und wurde im Abschnitt mit den Voraussetzungen eingerichtet. Sie müssen diese Informationen nicht direkt in Ihr Skript eingeben.

3. Optional – Stellen Sie die `transformation_ctx`-Parameter der Methode bereit, um Auftragslesezeichen zu unterstützen. Im folgenden Abschnitt the section called “Optional – Auftragslesezeichen aktivieren” erfahren Sie mehr über Auftragslesezeichen.

> **Note**
>
> Gängige Methoden zum Extrahieren von Daten
> the section called “create_dynamic_frame_from_catalog”wird verwendet, um eine Verbindung zu Tabellen im AWS Glue-Datenkatalog herzustellen.
> Wenn Sie Ihren Auftrag direkt mit einer Konfiguration versehen müssen, die die Struktur und den Standort Ihrer Quelle beschreibt, sehen Sie sich die Methode the section called “create_dynamic_frame_from_options” an. Sie müssen detailliertere Parameter zur Beschreibung Ihrer Daten angeben als bei `create_dynamic_frame.from_catalog`. Weitere Informationen zur Identifikation der erforderlichen Parameter finden Sie in der zusätzlichen Dokumentation zu `format_options` und `connection_parameters`. Eine Erläuterung, wie Sie Ihre Skriptinformationen zu Ihrem Quelldatenformat bereitstellen, finden Sie unter the section called “Pfad-Formatoptionen”. Eine Erläuterung, wie Sie Ihre Skriptinformationen zu Ihrem Quelldatenstandort bereitstellen, finden Sie unter the section called “Verbindungsparameter”.
> Wenn Sie Informationen aus einer Streaming-Quelle lesen, stellen Sie Ihrem Auftrag Quellinformationen über die Methoden the section called “create_data_frame_from_catalog” oder the section called “create_data_frame_from_options” bereit. Beachten Sie, dass diese Methoden Apache-Spark-`DataFrames` zurückgeben.
> Unser generierter Code ruft `create_dynamic_frame.from_catalog` auf, während sich die Referenzdokumentation auf `create_dynamic_frame_from_catalog` bezieht. Diese Methoden rufen letztendlich denselben Code auf und sind enthalten, sodass Sie saubereren

Code schreiben können. Sie können dies überprüfen, indem Sie sich die Quelle für unseren Python-Wrapper ansehen, verfügbar unter aws-glue-libs.

Schritt 4. Transformieren von Daten mit AWS Glue

Nach dem Extrahieren der Quelldaten in einem ETL-Prozess müssen Sie beschreiben, wie Sie Ihre Daten ändern möchten. Sie stellen diese Informationen bereit, indem Sie im visuellen Editor von AWS Glue Studio einen Transform-Knoten erstellen.

In diesem Schritt stellen Sie der `ApplyMapping`-Methode eine Karte mit aktuellen und gewünschten Feldnamen und -typen zur Transformation Ihres `DynamicFrame` bereit.

Führen Sie die folgenden Transformationen aus.

- Löschen Sie die vier `location`- und `province`-Schlüssel.
- Ändern Sie den Namen von `officer` zu `officer_name`.
- Ändern Sie den Typ von `ticket_number` und `set_fine_amount` zu `float`.

`create_dynamic_frame.from_catalog` bietet Ihnen ein `DynamicFrame`-Objekt. A `DynamicFrame` steht für einen Datensatz in AWS Glue. AWS Klebetransformationen sind Operationen, die sich ändern`DynamicFrames`.

> **Note**
>
> Was ist ein `DynamicFrame`?
> Ein `DynamicFrame` ist eine Abstraktion, mit der Sie einen Datensatz mit einer Beschreibung der Namen und Typen von Einträgen in den Daten verbinden können. In Apache Spark gibt es eine ähnliche Abstraktion, die als a bezeichnet wird. DataFrame Eine Erklärung von DataFrames finden Sie im Spark SQL Guide.
> Mit `DynamicFrames` können Sie Datensatzschemata dynamisch beschreiben. Stellen Sie sich einen Datensatz mit einer Preisspalte vor, in der einige Einträge den Preis als Zeichenfolge und andere den Preis als Doppelzahl speichern. AWS Glue berechnet ein Schema on-the-fly — es erstellt für jede Zeile einen sich selbst beschreibenden Datensatz. Inkonsistente Felder (wie Preis) werden explizit mit einem Typ dargestellt (`ChoiceType`) im Schema für den Rahmen. Sie können Ihre inkonsistenten Felder beheben, indem Sie sie mit `DropFields` löschen oder mit `ResolveChoice` auflösen. Dies sind

Transformationen, die auf dem `DynamicFrame` verfügbar sind. Sie können Ihre Daten dann mit `writeDynamicFrame` zurück in Ihren Data Lake schreiben.
Sie können viele der gleichen Transformationen über Methoden in der `DynamicFrame`-Klasse abrufen, was zu besser lesbaren Skripten führen kann. Mehr über `DynamicFrame` erfahren Sie unter the section called “DynamicFrame”.

In diesem Verfahren schreiben Sie den folgenden Code mithilfe von `ApplyMapping`. Dieser Code ist ein Teil des generierten Beispielskripts.

```
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)
```

Um Daten mit AWS Glue zu transformieren

1. Sehen Sie sich die Dokumentation an, um eine Transformation zum Ändern und Löschen von Feldern zu identifizieren. Details hierzu finden Sie unter the section called “GlueTransform”. Wählen Sie die Transformation `ApplyMapping` aus. Mehr über `ApplyMapping` erfahren Sie unter the section called “ApplyMapping”. Rufen Sie `apply` über das Transformationsobjekt `ApplyMapping` auf.

   > **Note**
   >
   > Was ist `ApplyMapping`?
   > `ApplyMapping` nimmt einen `DynamicFrame` und transformiert ihn. Es braucht eine Liste von Tupeln, die Transformationen auf Feldern darstellen – eine „Zuordnung“. Die ersten beiden Tupelelemente, ein Feldname und ein Typ, werden verwendet, um ein

Feld im Rahmen zu identifizieren. Die zweiten beiden Parameter sind ebenfalls ein Feldname und ein Feldtyp.
ApplyMapping konvertiert das Quellfeld in den Zielnamen und gibt einen neuen ein`DynamicFrame`, den es zurückgibt. Nicht bereitgestellte Felder werden im Rückgabewert gelöscht.
Statt `apply` aufzurufen, können Sie dieselbe Transform mit der `apply_mapping`-Methode im `DynamicFrame` aufrufen, um fließenderen, lesbareren Code zu erstellen. Weitere Informationen finden Sie unter the section called “apply_mapping”.

2. Sehen Sie sich die Dokumentation zu `ApplyMapping` an, um erforderliche Parameter zu identifizieren. Siehe the section called “ApplyMapping”. Sie werden feststellen, dass diese Methode die Parameter `frame` und `mappings` erfordert. Stellen Sie die notwendigen Parameter für `ApplyMapping` bereit.
3. Optional – Stellen Sie der Methode `transformation_ctx` bereit, um Auftragslesezeichen zu unterstützen. Im folgenden Abschnitt the section called “Optional – Auftragslesezeichen aktivieren” erfahren Sie mehr über Auftragslesezeichen.

> **Note**
>
> Apache-Spark-Funktion
> Wir bieten Transformationen, um ETL-Workflows in Ihrem Auftrag zu optimieren. Sie haben auch Zugriff auf die Bibliotheken, die in einem Spark-Programm in Ihrem Auftrag verfügbar sind, das für allgemeinere Zwecke entwickelt wurde. Um diese zu verwenden, konvertieren Sie zwischen `DynamicFrame` und `DataFrame`.
> Sie können einen `DataFrame` mit the section called “toDF” erstellen. Anschließend können Sie die auf der Website verfügbaren Methoden verwenden DataFrame , um Ihren Datensatz zu transformieren. Weitere Informationen zu diesen Methoden finden Sie unter DataFrame.
> Sie können dann rückwärts konvertierenthe section called “fromDF”, um Ihren Frame mithilfe von AWS Glue-Operationen auf ein Ziel zu laden.

Schritt 5. Daten in ein Ziel laden

Nachdem Sie Ihre Daten transformiert haben, speichern Sie die transformierten Daten in der Regel an einem anderen Ort als der Quelle. Sie führen diesen Vorgang durch, indem Sie im Visual Editor von AWS Glue Studio einen Zielknoten erstellen.

In diesem Schritt stellen Sie der `write_dynamic_frame.from_options`-Methode ein `connection_type`, `connection_options`, `format` und `format_options` bereit, um Daten in ein Ziel-Bucket in Amazon S3 zu laden.

Im ersten Schritt haben Sie ein `GlueContext`-Objekt initialisiert. In AWS Glue finden Sie hier Methoden, die zur Konfiguration von Zielen verwendet werden, ähnlich wie Quellen.

In diesem Verfahren schreiben Sie den folgenden Code mithilfe von `write_dynamic_frame.from_options`. Dieser Code ist ein Teil des generierten Beispielskripts.

```
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://amzn-s3-demo-bucket", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
    )
```

So laden Sie Daten in ein Ziel

1. Überprüfen Sie die Dokumentation, um eine Methode zum Laden von Daten in einen Ziel-Bucket von Amazon S3 zu finden. Diese Methoden sind dokumentiert in the section called “GlueContext”. Wählen Sie die the section called “write_dynamic_frame_from_options”-Methode. Rufen Sie diese Methode auf `glueContext` auf.

   > **Note**
   >
   > Gängige Methoden für das Laden von Daten
   > `write_dynamic_frame.from_options` ist die gängigste Methode zum Laden von Daten. Es unterstützt alle Ziele, die in AWS Glue verfügbar sind.
   > Wenn Sie in ein JDBC-Ziel schreiben, das in einer AWS Glue-Verbindung definiert ist, verwenden Sie die the section called “write_dynamic_frame_from_jdbc_conf” Methode. AWS In Klebeverbindungen werden Informationen darüber gespeichert, wie eine Verbindung zu einer Datenquelle hergestellt wird. Dadurch entfällt die Notwendigkeit, diese Informationen in `connection_options` bereitzustellen. Sie müssen jedoch immer noch `connection_options` verwenden, um `dbtable` bereitzustellen.
   > `write_dynamic_frame.from_catalog` ist keine gängige Methode zum Laden von Daten. Diese Methode aktualisiert den AWS Glue-Datenkatalog, ohne den zugrunde

liegenden Datensatz zu aktualisieren, und wird in Kombination mit anderen Prozessen verwendet, die den zugrunde liegenden Datensatz ändern. Weitere Informationen finden Sie unter the section called “Aktualisierung des Schemas und Hinzufügen neuer Partitionen”.

2. Untersuchen Sie die Dokumentation auf the section called “write_dynamic_frame_from_options”. Diese Methode erfordert `frame`, `connection_type`, `format`, `connection_options` und `format_options`. Rufen Sie diese Methode auf `glueContext` auf.

   a. Weitere Informationen zu den erforderlichen Parametern finden Sie in der zusätzlichen Dokumentation zu `format_options` und `format`. Eine Erläuterung der Datenformate finden Sie unter the section called “Pfad-Formatoptionen”.

   b. Weitere Informationen zu den erforderlichen Parametern finden Sie in der zusätzlichen Dokumentation zu `connection_type` und `connection_options`. Eine Erläuterung der Zusammenhänge finden Sie unter the section called “Verbindungsparameter”.

   c. Stellen Sie die notwendigen Parameter für `write_dynamic_frame.from_options` bereit. Diese Methode hat eine ähnliche Konfiguration wie `create_dynamic_frame.from_options`.

3. Optional – Stellen Sie `transformation_ctx` für `write_dynamic_frame.from_options` bereit, um Auftragslesezeichen zu unterstützen. Im folgenden Abschnitt the section called “Optional – Auftragslesezeichen aktivieren” erfahren Sie mehr über Auftragslesezeichen.

Schritt 6: Bestätigen des **`Job`**-Objekts

In Schritt 1 haben Sie ein `Job`-Objekt initialisiert. Möglicherweise müssen Sie den Lebenszyklus am Ende Ihres Skripts manuell abschließen, wenn dies für bestimmte optionale Funktionen erforderlich ist, um ordnungsgemäß zu funktionieren, z. B. bei der Verwendung von Job-Lesezeichen. Diese Arbeit wird hinter den Kulissen von AWS Glue Studio erledigt.

Rufen Sie in diesem Schritt die `commit`-Methode für das `Job`-Objekt auf.

In diesem Verfahren schreiben Sie den folgenden Code. Dieser Code ist ein Teil des generierten Beispielskripts.

```
job.commit()
```

So bestätigen Sie das **`Job`**-Objekt

1. Wenn Sie dies noch nicht getan haben, führen Sie die optionalen Schritte aus, die in den vorherigen Abschnitten beschrieben wurden, um `transformation_ctx` einzubeziehen.
2. Rufen Sie die folgende Seite auf `commit`.

Optional – Auftragslesezeichen aktivieren

In jedem vorherigen Schritt wurden Sie angewiesen, `transformation_ctx`-Parameter einzurichten. Dies bezieht sich auf ein Feature namens Auftragslesezeichen.

Mit Auftragslesezeichen können Sie bei Aufträgen, die auf wiederkehrender Basis ausgeführt werden, Zeit und Geld sparen, indem Sie auf Datensätze zugreifen, bei denen frühere Arbeiten leicht nachverfolgt werden können. Job-Lesezeichen verfolgen den Fortschritt einer AWS Glue-Transformation in einem Datensatz aus früheren Durchläufen. Indem AWS Glue nachverfolgt, wo frühere Läufe geendet haben, kann es seine Arbeit auf Zeilen beschränken, die es zuvor nicht verarbeitet hat. Weitere Informationen zu Auftragslesezeichen finden Sie unter the section called “Verfolgen von verarbeiteten Daten mit Auftragslesezeichen”.

Um Auftragslesezeichen zu aktivieren, fügen Sie zunächst die `transformation_ctx`-Anweisungen in unsere bereitgestellten Funktionen ein, wie in den vorherigen Beispielen beschrieben.
Der Status des Auftragslesezeichens wird über mehrere Ausführungen hinweg beibehalten. `transformation_ctx`-Parameter sind Schlüssel, die für den Zugriff auf diesen Status verwendet werden. Für sich genommen werden diese Aussagen nichts bewirken. Sie müssen das Feature auch in der Konfiguration für Ihren Auftrag aktivieren.

In diesem Verfahren aktivieren Sie Auftragslesezeichen mit dem AWS Management Console.

So aktivieren Sie Auftragslesezeichen

1. Navigieren Sie zum Abschnitt Job details (Auftragsdetails) Ihres entsprechenden Auftrags.
2. Legen Sie für Job bookmark (Auftragslesezeichen) Enable (Aktivieren) fest.

Schritt 7. Ausführen Ihres Codes als Auftrag

In diesem Schritt führen Sie Ihren Auftrag aus, um zu überprüfen, ob Sie dieses Tutorial erfolgreich abgeschlossen haben. Dies erfolgt mit einem Klick auf eine Schaltfläche, wie im visuellen Editor von AWS Glue Studio.

So führen Sie Ihren Code als Auftrag aus

1. Wählen Sie in der Titelleiste Untitled Job (Auftrag ohne Titel) aus, um Ihren Auftragsnamen zu bearbeiten und festzulegen.
2. Navigieren Sie zur Registerkarte Job details (Auftragsdetails). Weisen Sie Ihrem Auftrag eine IAM-Rolle zu. Sie können die mit der AWS CloudFormation Vorlage erstellte Datei in den Voraussetzungen für das AWS Glue Studio-Tutorial verwenden. Wenn Sie dieses Tutorial abgeschlossen haben, sollte es verfügbar sein als `AWS Glue StudioRole`.
3. Wählen Sie Save (Speichern), um Ihr Skript zu speichern.
4. Wählen Sie Run (Ausführen) aus, um Ihren Auftrag auszuführen.
5. Navigieren Sie zur Registerkarte Runs (Ausführungen), um zu überprüfen, ob Ihr Auftrag abgeschlossen ist.
6. Navigieren Sie zu*`amzn-s3-demo-bucket`*, dem Ziel für`write_dynamic_frame.from_options`. Bestätigen Sie, dass die Ausgabe Ihren Erwartungen entspricht.

Weitere Informationen zum Konfigurieren und Verwalten von Aufträgen finden Sie unter the section called “Bereitstellen eigener, benutzerdefinierter Skripts”.

Weitere Informationen

Apache Spark-Bibliotheken und -Methoden sind in AWS Glue-Skripten verfügbar. Sie können sich die Spark-Dokumentation ansehen, um zu verstehen, was Sie mit diesen enthaltenen Bibliotheken tun können. Weitere Informationen finden Sie im Abschnitt über Beispiele des Spark-Quell-Repositorys.

AWS Glue 2.0+ enthält standardmäßig mehrere gängige Python-Bibliotheken. Es gibt auch Mechanismen zum Laden eigener Abhängigkeiten in einen AWS Glue-Job in einer Scala- oder Python-Umgebung. Weitere Informationen zu Python-Abhängigkeiten finden Sie unter the section called “Python-Bibliotheken”.

Weitere Beispiele für die Verwendung von AWS Glue-Funktionen in Python finden Sie unterthe section called “Python-Beispiele”. Scala- und Python-Aufträge haben Feature-Parität, daher sollten unsere Python-Beispiele Ihnen einen Eindruck davon geben, wie Sie ähnliche Aufgaben in Scala ausführen können.

Programmieren AWS Sie Glue ETL-Skripte in PySpark

Python-Codebeispiele und Hilfsprogramme für AWS Glue finden Sie im AWS Glue-Beispiel-Repository auf der GitHub Website.

Python mit AWS Glue verwenden

AWS Glue unterstützt eine Erweiterung des PySpark Python-Dialekts für Scripting-Extraktions-, Transformations- und Ladeaufträge (ETL). In diesem Abschnitt wird beschrieben, wie Python in ETL-Skripten und mit der AWS Glue-API verwendet wird.

- Einrichtung für die Verwendung von Python mit AWS Glue
- Calling (Anrufen) AWS Glue APIs in Python
- Python-Bibliotheken mit AWS Glue verwenden
- AWS Glue Python-Codebeispiele

AWS PySpark Verlängerungen Glue

AWS Glue hat die folgenden Erweiterungen für den PySpark Python-Dialekt erstellt.

- Zugriff auf Parameter mit `getResolvedOptions`
- PySpark Erweiterungstypen
- DynamicFrame Klasse
- DynamicFrameCollection Klasse
- DynamicFrameWriter Klasse
- DynamicFrameReader Klasse
- GlueContext Klasse

AWS PySpark Klebetransformationen

AWS Glue hat die folgenden Transformationsklassen für die Verwendung in PySpark ETL-Vorgängen erstellt.

- GlueTransform Basisklasse
- ApplyMapping Klasse
- DropFields Klasse

- DropNullFields Klasse
- ErrorsAsDynamicFrame Klasse
- FillMissingValues Klasse
- Filterklasse
- FindIncrementalMatches Klasse
- FindMatches Klasse
- FlatMap Klasse
- Join-Klasse
- Map-Klasse
- MapToCollection Klasse
- mergeDynamicFrame
- Relationalize-Klasse
- RenameField Klasse
- ResolveChoice Klasse
- SelectFields Klasse
- SelectFromCollection Klasse
- Spigot-Klasse
- SplitFields Klasse
- SplitRows Klasse
- Unbox-Klasse
- UnnestFrame Klasse

Einrichtung für die Verwendung von Python mit AWS Glue

Verwenden Sie Python, um Ihre ETL-Skripte für Spark-Aufträge zu entwickeln. Die unterstützten Python-Versionen für ETL-Jobs hängen von AWS Glue Version des Jobs. Weitere Informationen zu AWS Glue Versionen finden Sie unterGlue version job property.

Um Ihr System für die Verwendung von Python einzurichten mit AWS Glue

Folgen Sie diesen Schritten, um Python zu installieren und das aufrufen zu können AWS Glue APIs.

1. Wenn Sie Python noch nicht installiert haben, laden Sie es von der Downloadseite unter Python.org herunter und installieren Sie es.

2. Installieren Sie AWS Command Line Interface (AWS CLI) wie in der AWS CLI-Dokumentation dokumentiert.

   Das AWS CLI ist für die Verwendung von Python nicht direkt erforderlich. Die Installation und Konfiguration ist jedoch eine bequeme Methode, um es AWS mit Ihren Kontoanmeldeinformationen einzurichten und zu überprüfen, ob sie funktionieren.
3. Installieren Sie das AWS SDK für Python (Boto 3), wie im Boto3-Schnellstart dokumentiert.

   Boto 3-Ressourcen APIs sind noch nicht verfügbar für AWS Glue. Derzeit kann nur der Boto 3-Client verwendet APIs werden.

   Weitere Informationen zu Boto 3 finden Sie unter AWS SDK for Python (Boto3) Getting Started (Erste Schritte mit SDK for Python (Boto3)).

Sie finden Python-Codebeispiele und Hilfsprogramme für AWS Glue in der AWS Glue Proben-Repository auf der GitHub Website.

Calling (Anrufen) AWS Glue APIs in Python

Beachten Sie, dass Boto 3-Ressourcen noch nicht verfügbar APIs sind für AWS Glue. Derzeit kann nur der Boto 3-Client verwendet APIs werden.

AWS Glue API-Namen in Python

AWS Glue-API-Namen in Java und anderen Programmiersprachen sind im Allgemeinen CamelCased. Wenn diese allgemeinen Namen jedoch von Python aus aufgerufen werden, werden sie in Kleinbuchstaben geändert und die einzelnen Teile des Namens werden durch Unterstriche getrennt, damit sie besser "an Python angepasst" sind. In der AWS Glue API Referenzdokumentation sind diese pythonischen Namen in Klammern hinter den generischen Namen aufgeführt. CamelCased

Obwohl die AWS Glue API-Namen selbst werden in Kleinbuchstaben umgewandelt, ihre Parameternamen bleiben jedoch großgeschrieben. Es ist wichtig, sich daran zu erinnern, da Parameter beim Aufrufen namentlich übergeben werden sollten AWS Glue APIs, wie im folgenden Abschnitt beschrieben.

Python-Parameter übergeben und darauf zugreifen in AWS Glue

In Python ruft man auf AWS Glue APIs, ist es am besten, Parameter explizit nach Namen zu übergeben. Zum Beispiel:

```
job = glue.create_job(Name='sample', Role='Glue_DefaultRole',
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'})
```

Es ist hilfreich, wenn Sie verstehen, dass Python ein Wörterbuch mit den Name-Wert-Tupeln erstellt, die Sie als Argumente für ein ETL-Skript in einem Auftrags-Struktur oder JobRun Struktur angeben. Boto 3 übergibt sie dann an AWS Glue im JSON-Format über einen REST-API-Aufruf. Das bedeutet, dass Sie sich nicht auf die Reihenfolge der Argumente verlassen können, wenn Sie in Ihrem Skript auf sie zugreifen.

Angenommen, Sie starten eine `JobRun` in einer Python-Lambda-Handler-Funktion und Sie möchten mehrere Parameter angeben. Ihr Code würde in etwa wie folgt aussehen:

```
from datetime import datetime, timedelta

client = boto3.client('glue')

def lambda_handler(event, context):
  last_hour_date_time = datetime.now() - timedelta(hours = 1)
  day_partition_value = last_hour_date_time.strftime("%Y-%m-%d")
  hour_partition_value = last_hour_date_time.strftime("%-H")

  response = client.start_job_run(
               JobName = 'my_test_Job',
               Arguments = {
                 '--day_partition_key':   'partition_0',
                 '--hour_partition_key':  'partition_1',
                 '--day_partition_value':  day_partition_value,
                 '--hour_partition_value': hour_partition_value } )
```

Um in Ihrem ETL-Skript zuverlässig auf diese Parameter zuzugreifen, geben Sie sie namentlich an mit AWS Gluedie `getResolvedOptions` Funktion und greifen Sie dann über das resultierende Wörterbuch auf sie zu:

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
```

```
                            'hour_partition_key',
                            'day_partition_value',
                            'hour_partition_value'])
print "The day partition key is: ", args['day_partition_key']
print "and the day partition value is: ", args['day_partition_value']
```

Wenn Sie ein Argument übergeben möchten, das eine verschachtelte JSON-Zeichenfolge ist, um den Parameterwert beizubehalten, wenn er an Ihr AWS Glue ETL-Job: Sie müssen die Parameterzeichenfolge kodieren, bevor Sie die Jobausführung starten, und dann die Parameterzeichenfolge dekodieren, bevor Sie sie in Ihrem Jobskript referenzieren. Betrachten wir z. B. die folgende Argumentzeichenfolge:

```
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": '{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
})
```

Um diesen Parameter korrekt zu übergeben, sollten Sie das Argument als Base64-codierte Zeichenfolge codieren.

```
import base64
...
sample_string='{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
sample_string_bytes = sample_string.encode("ascii")

base64_bytes = base64.b64encode(sample_string_bytes)
base64_string = base64_bytes.decode("ascii")
...
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": base64_bytes})
...
sample_string_bytes = base64.b64decode(base64_bytes)
sample_string = sample_string_bytes.decode("ascii")
print(f"Decoded string: {sample_string}")
...
```

Beispiel: Erstellen und Ausführen eines Auftrags

Das folgende Beispiel zeigt, wie der Aufruf von AWS Glue APIsVerwenden von Python, um einen ETL-Job zu erstellen und auszuführen.

So erstellen Sie einen Auftrag und führen ihn aus

1. Erstellen Sie eine Instanz von AWS Glue Kunde:

```
import boto3
glue = boto3.client(service_name='glue', region_name='us-east-1',
              endpoint_url='https://glue.us-east-1.amazonaws.com')
```

2. Erstellen Sie einen Auftrag. Sie müssen `glueetl` als Namen für den ETL-Befehl verwenden, wie im folgenden Code gezeigt:

```
myJob = glue.create_job(Name='sample', Role='Glue_DefaultRole',
                        Command={'Name': 'glueetl',
                                 'ScriptLocation': 's3://my_script_bucket/
scripts/my_etl_script.py'})
```

3. Starten Sie eine neue Ausführung des Auftrags, den Sie im vorherigen Schritt erstellt haben:

```
myNewJobRun = glue.start_job_run(JobName=myJob['Name'])
```

4. Rufen Sie den Auftragsstatus ab:

```
status = glue.get_job_run(JobName=myJob['Name'], RunId=myNewJobRun['JobRunId'])
```

5. Drucken Sie den aktuellen Status der Auftragsausführung:

```
print(status['JobRun']['JobRunState'])
```

Python-Bibliotheken mit AWS Glue verwenden

AWS Mit Glue können Sie zusätzliche Python-Module und -Bibliotheken für die Verwendung mit AWS Glue ETL installieren.

Themen

- Installation zusätzlicher Python-Bibliotheken in AWS Glue 5.0 mit requirements.txt
- Installation zusätzlicher Python-Module mit Pip in AWS Glue 2.0 oder höher
- Einschließlich Python-Dateien mit PySpark nativen Funktionen
- Programmierskripten, die visuelle Transformationen verwenden
- Python-Module sind bereits in AWS Glue enthalten

- Komprimieren von Bibliotheken für die Integration
- Python-Bibliotheken in AWS Glue Studio-Notebooks laden
- Laden von Python-Bibliotheken in einen Entwicklungsendpunkt
- Python-Bibliotheken in einem Job verwenden oder JobRun

Installation zusätzlicher Python-Bibliotheken in AWS Glue 5.0 mit requirements.txt

In AWS Glue 5.0 können Sie den Defacto-Standard zur Verwaltung von `requirements.txt` Python-Bibliotheksabhängigkeiten bereitstellen. Geben Sie dazu die folgenden zwei Job-Parameter an:

- Schlüssel: `--python-modules-installer-option`

  Wert: `-r`
- Schlüssel: `--additional-python-modules`

  Wert: `s3://path_to_requirements.txt`

AWS Glue 5.0-Knoten laden zunächst die in angegebenen Python-Bibliotheken`requirements.txt`. Hier ist ein Beispiel für requirements.txt:

```
awswrangler==3.9.1
elasticsearch==8.15.1
PyAthena==3.9.0
PyMySQL==1.1.1
PyYAML==6.0.2
pyodbc==5.2.0
pyorc==0.9.0
redshift-connector==2.1.3
scipy==1.14.1
scikit-learn==1.5.2
SQLAlchemy==2.0.36
```

Installation zusätzlicher Python-Module mit Pip in AWS Glue 2.0 oder höher

AWS Glue verwendet den Python Package Installer (pip3), um zusätzliche Module zu installieren, die von AWS Glue ETL verwendet werden sollen. Sie können den Parameter „`--additional-python-modules`“ mit verschiedenen kommagetrennten Python-Modulen verwenden, um ein neues Modul hinzuzufügen oder die Version eines vorhandenen Moduls zu ändern. Sie können

benutzerdefinierte Distributionen einer Bibliothek installieren, indem Sie die Distribution in Amazon S3 hochladen und dann den Pfad zum Amazon-S3-Objekt in Ihre Modulliste aufnehmen.

Sie können zusätzliche Optionen an pip3 übergeben mit dem Parameter `--python-modules-installer-option`. Sie können z. B. `"--upgrade"` übergeben, um die Pakete zu aktualisieren, die von `"--additional-python-modules"` angegeben wurden. Weitere Beispiele finden Sie unter Python-Module aus einem Rad für Spark-ETL-Workloads mit AWS Glue 2.0 erstellen.

Wenn Ihre Python-Abhängigkeiten transitiv von nativem, kompiliertem Code abhängen, können Sie gegen die folgende Einschränkung verstoßen: AWS Glue unterstützt das Kompilieren von nativem Code in der Jobumgebung nicht. AWS Glue-Jobs werden jedoch in einer Amazon Linux 2-Umgebung ausgeführt. Möglicherweise können Sie Ihre nativen Abhängigkeiten in einer kompilierten Form über einen Wheel-Verteiler bereitstellen.

Verwenden Sie zum Aktualisieren oder Hinzufügen eines neuen `scikit-learn`-Moduls den folgenden Schlüssel/Wert: `"--additional-python-modules", "scikit-learn==0.21.3"`.

Auch innerhalb der Option `--additional-python-modules` können Sie einen Amazon-S3-Pfad zu einem Python-Wheel-Modul angeben. Zum Beispiel:

```
--additional-python-modules s3://aws-glue-native-spark/tests/j4.2/ephem-3.7.7.1-cp37-cp37m-linux_x86_64.whl,s3://aws-glue-native-spark/tests/j4.2/fbprophet-0.6-py3-none-any.whl,scikit-learn==0.21.3
```

Sie geben das `--additional-python-modules` im Feld Job-Parameter der AWS Glue-Konsole an oder ändern die Job-Argumente im AWS SDK. Informationen zum Festlegen von Auftragsparametern finden Sie unter the section called “Auftragsparameter”.

Einschließlich Python-Dateien mit PySpark nativen Funktionen

AWS Glue verwendet PySpark , um Python-Dateien in AWS Glue ETL-Jobs einzubeziehen. Sie sollten `--additional-python-modules` verwenden, um Ihre Abhängigkeiten zu verwalten (sofern verfügbar). Sie können den Auftragsparameter `--extra-py-files` verwenden, um Python-Dateien hinzuzufügen. Abhängigkeiten müssen in Amazon S3 gehostet werden und der Argumentwert sollte eine kommagetrennte Liste von Amazon S3-Pfaden ohne Leerzeichen sein. Diese Funktion verhält sich wie das Python-Abhängigkeitsmanagement, das Sie mit Spark verwenden würden´. Weitere Informationen zur Python-Abhängigkeitsverwaltung in Spark finden Sie auf der Seite PySpark Using Native Features in der Apache Spark-Dokumentation. `--extra-py-files`ist nützlich in Fällen, in denen Ihr zusätzlicher Code nicht gepackt ist oder wenn Sie ein Spark-Programm mit einer

vorhandenen Toolchain zur Verwaltung von Abhängigkeiten migrieren. Damit Ihre Abhängigkeitstools wartbar sind, müssen Sie Ihre Abhängigkeiten bündeln, bevor Sie sie einreichen.

Programmierskripten, die visuelle Transformationen verwenden

Wenn Sie einen AWS Glue-Job mit der visuellen Oberfläche von AWS Glue Studio erstellen, können Sie Ihre Daten mit verwalteten Datentransformationsknoten und benutzerdefinierten visuellen Transformationen transformieren. Weitere Informationen zu verwalteten Datentransformationsknoten finden Sie unterthe section called “Transformieren Sie Daten mit AWS Glue verwaltete Transformationen”. Weitere Informationen zu benutzerdefinierten visuellen Transformationen finden Sie unterthe section called “ Transformieren Sie Daten mit benutzerdefinierten visuellen Transformationen ”. Skripten, die visuelle Transformationen verwenden, können nur generiert werden, wenn Ihre Jobsprache auf Python eingestellt ist.

Wenn ein AWS Glue-Job mithilfe von visuellen Transformationen generiert wird, AWS bezieht Glue Studio diese Transformationen mithilfe des `--extra-py-files` Parameters in der Auftragskonfiguration in die Laufzeitumgebung ein. Informationen zu Auftragsparametern finden Sie unter the section called “Auftragsparameter”. Wenn Sie Änderungen an einem generierten Skript oder einer generierten Laufzeitumgebung vornehmen, müssen Sie diese Jobkonfiguration beibehalten, damit Ihr Skript erfolgreich ausgeführt werden kann.

Python-Module sind bereits in AWS Glue enthalten

Um die Version dieser bereitgestellten Module zu ändern, stellen Sie neue Versionen mit dem Auftragsparameter `--additional-python-modules` bereit.

AWS Glue version 5.0

AWS Glue Version 5.0 enthält standardmäßig die folgenden Python-Module:

- aiobotocore==2.13.1
- Aio Happy Eyeballs ==2.3.5
- aiohttp==3.10.1
- aioitertools==0.11.0
- aiosignal==1.3.1
- Anwendungsverzeichnis==1.4.4
- attrs==24.2.0
- Boot 3 = 1.34.131

- Botocore==1.34.131
- Zertifikat == 2024.7.4
- Zeichensatz-Normalisierer==3.3.2
- Konturpie==1.2.1
- Radfahrer ==0.12.1
- Schriftwerkzeuge==4.53.1
- eingefrorene Liste==1.4.1
- fsspec==2024.6.1
- idna==2.10
- jmespath==0.10.0
- Kaleido==0.2.1
- Kiwi-Solver==1.4.5
- matplotlib==3.9.0
- Multidict==6.0.5
- numpy==1.26.4
- Verpackung==24.1
- Pandas==2.2.2
- Kissen==10.4.0
- pip==22.3.1
- Handlung == 5.23.0
- Pfeil==17.0.0
- Pyparsing==3.1.2
- python-dateutil==2.9.0.post0
- pytz==2024.1
- Anfragen ==2.32.2
- s3fs==2024.6.1
- s3-Übertragung ==0.10.2
- in der See geboren==0.13.2
- Einrichtungswerkzeuge == 59.6.0
- six==1.16.0

- Hartnäckigkeit==9.0.0
- tzdata == 2024.1
- urllib3==1.25.10
- virtuelle Umgebung == 20.4.0
- Wrapt==1.16.0
- Yarl==1.9.4

AWS Glue version 4.0

AWS Glue Version 4.0 enthält standardmäßig die folgenden Python-Module:

- aiobotocore==2.4.1
- aiohttp==3.8.3
- aioitertools==0.11.0
- aiosignal==1.3.1
- async-timeout==4.0.2
- asynctest==0.13.0
- attrs==22.2.0
- avro-python3==1.10.2
- boto3==1.24.70
- botocore==1.27.59
- certifi==2021.5.30
- chardet 3.0.4
- charset-normalizer==2.1.1
- click==8.1.3
- cycler 0.10.0
- Cython==0.29.32
- docutils==0.17.1
- enum34==1.1.10
- frozenlist==1.3.3
- fsspec==2021.8.1

- idna==2.10
- importlib-metadata==5.0.0
- jmespath==0.10.0
- joblib==1.0.1
- kaleido==0.2.1
- kiwisolver==1.4.4
- matplotlib==3.4.3
- mpmath==1.2.1
- multidict==6.0.4
- nltk==3.7
- numpy==1.23.5
- packaging==23.0
- pandas==1.5.1
- patsy 0.5.1
- Pillow==9.4.0
- pip==23.0.1
- Handlung == 5.16.0
- pmdarima==2.0.1
- ptvsd 4.3.2
- pyarrow==10.0.0
- pydevd==2.5.0
- pyhocon==0.3.58
- PyMySQL==1.0.2
- pyparsing==2.4.7
- python-dateutil==2.8.2
- pytz==2021.1
- PyYAML==6.0.1
- regex==2022.10.31
- requests 2.23.0
- s3fs==2022.11.0

- s3transfer==0.6.0
- scikit-learn==1.1.3
- scipy==1.9.3
- setuptools==49.1.3
- six==1.16.0
- statsmodels==0.13.5
- subprocess32 3.5.4
- sympy==1.8
- tbats==1.1.0
- threadpoolctl==3.1.0
- tqdm==4.64.1
- typing_extensions==4.4.0
- urllib3==1.25.11
- wheel==0.37.0
- wrapt==1.14.1
- yarl==1.8.2
- zipp==3.10.0

AWS Glue version 3.0

AWS Glue Version 3.0 enthält standardmäßig die folgenden Python-Module:

- aiobotocore==1.4.2
- aiohttp==3.8.3
- aioitertools==0.11.0
- aiosignal==1.3.1
- async-timeout==4.0.2
- asynctest==0.13.0
- attrs==22.2.0
- avro-python3==1.10.2
- boto3==1.18.50

- botocore==1.21.50
- certifi==2021.5.30
- chardet 3.0.4
- charset-normalizer==2.1.1
- click==8.1.3
- cycler 0.10.0
- Cython==0.29.4
- docutils==0.17.1
- enum34==1.1.10
- frozenlist==1.3.3
- fsspec==2021.8.1
- idna==2.10
- importlib-metadata==6.0.0
- jmespath==0.10.0
- joblib==1.0.1
- kiwisolver==1.3.2
- matplotlib==3.4.3
- mpmath==1.2.1
- multidict==6.0.4
- nltk==3.6.3
- numpy==1.19.5
- packaging==23.0
- pandas==1.3.2
- patsy 0.5.1
- Pillow==9.4.0
- pip==23.0
- pmdarima==1.8.2
- ptvsd 4.3.2
- pyarrow==5.0.0
- pydevd==2.5.0

- pyhocon==0.3.58
- PyMySQL==1.0.2
- pyparsing==2.4.7
- python-dateutil==2.8.2
- pytz==2021.1
- PyYAML==5.4.1
- regex==2022.10.31
- requests 2.23.0
- s3fs==2021.8.1
- s3transfer==0.5.0
- scikit-learn==0.24.2
- scipy==1.7.1
- six==1.16.0
- Spark==1.0
- statsmodels==0.12.2
- subprocess32 3.5.4
- sympy==1.8
- tbats==1.1.0
- threadpoolctl==3.1.0
- tqdm==4.64.1
- typing_extensions==4.4.0
- urllib3==1.25.11
- wheel==0.37.0
- wrapt==1.14.1
- yarl==1.8.2
- zipp==3.12.0

AWS Glue version 2.0

AWS Glue Version 2.0 enthält standardmäßig die folgenden Python-Module:

- avro-python3==1.10.0
- awscli==1.27.60
- boto3 1.12.4
- botocore 1.15.4
- certifi 2019.11.28
- chardet 3.0.4
- click==8.1.3
- colorama==0.4.4
- cycler 0.10.0
- Cython 0.29.15
- docutils 0.15.2
- enum34 1.1.9
- fsspec 0.6.2
- idna 2.9
- importlib-metadata==6.0.0
- jmespath 0.9.4
- joblib 0.14.1
- kiwisolver 1.1.0
- matplotlib 3.1.3
- mpmath 1.1.0
- nltk==3.5
- numpy 1.18.1
- pandas 1.0.1
- patsy 0.5.1
- pmdarima 1.5.3
- ptvsd 4.3.2
- pyarrow 0.16.0
- pyasn1==0.4.8
- pydevd 1.9.0
- pyhocon 0.3.54

- PyMySQL==0.9.3
- pyparsing 2.4.6
- python-dateutil==2.8.1
- pytz 2019.3
- PyYAML==5.3.1
- regex==2022.10.31
- requests 2.23.0
- rsa==4.7.2
- s3fs 0.4.0
- s3transfer 0.3.3
- scikit-learn 0.22.1
- scipy 1.4.1
- setuptools 45.2.0
- six 1.14.0
- Spark==1.0
- statsmodels 0.11.1
- subprocess32 3.5.4
- sympy 1.5.1
- tbats 1.0.9
- tqdm==4.64.1
- typing-extensions==4.4.0
- urllib3 1.25.8
- wheel==0.35.1
- zipp==3.12.0

Komprimieren von Bibliotheken für die Integration

Sofern eine Bibliothek nicht in einer einzigen `.py`-Datei enthalten ist, sollte sie in ein `.zip`-Archiv gepackt werden. Das Paketverzeichnis sollte sich im Stammverzeichnis des Archivs befinden und eine `__init__.py`-Datei für das Paket enthalten. Python ist dann in der Lage, das Paket wie gewohnt zu importieren.

Wenn Ihre Bibliothek nur aus einem einzigen Python-Modul in einer `.py`-Datei besteht, brauchen Sie sie nicht in einer `.zip`-Datei abzulegen.

Python-Bibliotheken in AWS Glue Studio-Notebooks laden

Informationen zur Angabe von Python-Bibliotheken in AWS Glue Studio-Notebooks finden Sie unter Zusätzliche Python-Module installieren.

Laden von Python-Bibliotheken in einen Entwicklungsendpunkt

Wenn Sie verschiedene Bibliotheks-Sets für verschiedene ETL-Skripts verwenden, können Sie entweder für jeden Satz einen eigenen Entwicklungsendpunkt einrichten oder die `.zip`-Bibliotheksdatei(en) überschreiben, die Ihr Entwicklungsendpunkt bei jedem Wechsel des Skripts lädt.

Sie können die Konsole verwenden, um eine oder mehrere Library.zip-Dateien für einen Entwicklungsendpunkt anzugeben, wenn Sie diesen erstellen. Nachdem Sie einen Namen und eine IAM-Rolle zugewiesen haben, wählen Sie Script Libraries and job parameters (optional) (Skript-Bibliotheken und Auftragsparameter (optional)) aus und geben Sie den vollständigen Amazon-S3-Pfad zu Ihrer `.zip`-Bibliotheksdatei im Feld Python library path (Python-Bibliothekspfad) ein. Zum Beispiel:

```
s3://bucket/prefix/site-packages.zip
```

Wenn Sie möchten, können Sie mehrere vollständige Pfade zu Dateien angeben und diese mit Kommas, aber ohne Leerzeichen trennen:

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Wenn Sie diese `.zip`-Dateien später aktualisieren, können Sie sie über die Konsole erneut in Ihren Entwicklungsendpunkt importieren. Navigieren Sie zu dem betreffenden Entwicklerendpunkt, markieren Sie das Kästchen daneben und wählen Sie Update ETL libraries (ETL-Bibliotheken aktualisieren) aus dem Menü Action (Aktion) aus.

Auf ähnliche Weise können Sie Bibliotheksdateien mit dem AWS Glue angeben APIs. Wenn Sie einen Entwicklungsendpunkt durch den Aufruf vonCreateDevEndpoint Aktion (Python: create_dev_endpoint) erstellen, können Sie einen oder mehrere vollständige Pfade zu Bibliotheken im `ExtraPythonLibsS3Path`-Parameter angeben, und zwar in einem Aufruf, der so aussieht:

```
dep = glue.create_dev_endpoint(
             EndpointName="testDevEndpoint",
             RoleArn="arn:aws:iam::123456789012",
             SecurityGroupIds="sg-7f5ad1ff",
             SubnetId="subnet-c12fdba4",
             PublicKey="ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCtp04H/y...",
             NumberOfNodes=3,
             ExtraPythonLibsS3Path="s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/
lib_X.zip")
```

Wenn Sie einen Entwicklungsendpunkt aktualisieren, können Sie auch die geladene Bibliotheken aktualisieren, indem Sie ein DevEndpointCustomLibraries-Objekt verwenden und den `UpdateEtlLibraries` -Parameter beim Aufruf von UpdateDevEndpoint (update_dev_endpoint) auf `True` festlegen.

Python-Bibliotheken in einem Job verwenden oder JobRun

Wenn Sie einen neuen Auftrag in der Konsole erstellen, können Sie eine oder mehrere Library.zip-Dateien angeben, indem Sie Script Libraries and job parameters (optional) (Skript-Bibliotheken und Auftragsparameter (optional)) auswählen und den vollständigen Amazon-S3-Bibliothekspfad wie beim Erstellen eines Entwicklungsendpunkts eingeben:

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Wenn Sie die Funktion CreateJob (Job erstellen) aufrufen, können Sie einen oder mehrere vollständige Pfade zu Standardbibliotheken angeben, indem Sie den `--extra-py-files`-Standardparameter verwenden:

```
job = glue.create_job(Name='sampleJob',
                      Role='Glue_DefaultRole',
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'},
                      DefaultArguments={'--extra-py-files': 's3://bucket/prefix/
lib_A.zip,s3://bucket_B/prefix/lib_X.zip'})
```

Wenn Sie dann eine starten JobRun, können Sie die Standardeinstellung der Bibliothek durch eine andere überschreiben:

```
runId = glue.start_job_run(JobName='sampleJob',
```

```
                        Arguments={'--extra-py-files': 's3://bucket/prefix/
lib_B.zip'})
```

AWS Glue Python-Codebeispiele

- Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten
- Codebeispiel: Datenvorbereitung mit ResolveChoice, Lambda und ApplyMapping

Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten

Dieses Beispiel verwendet einen Datensatz, der von http://everypolitician.org/ in den `sample-dataset` Bucket in Amazon Simple Storage Service (Amazon S3) heruntergeladen wurde: `s3://awsglue-datasets/examples/us-legislators/all`. Das Dataset enthält Daten im JSON-Format zu Gesetzgebern der Vereinigten Staaten und den Sitzen, die sie im US-Repräsentantenhaus und Senat innegehabt haben. Diese Daten wurden zum Zwecke dieses Tutorials geringfügig geändert und in einem öffentlichen Amazon S3-Bucket verfügbar gemacht.

Den Quellcode für dieses Beispiel finden Sie in der `join_and_relationalize.py` Datei im AWS Glue Beispiel-Repository auf der GitHub Website.

Anhand dieser Daten wird in diesem Tutorial veranschaulicht, wie Sie die folgenden Aufgaben ausführen:

- Benutze ein AWS Glue Crawler zum Klassifizieren von Objekten, die in einem öffentlichen Amazon S3 S3-Bucket gespeichert sind, und zum Speichern ihrer Schemas im AWS Glue-Datenkatalog.
- Untersuchen Sie die Tabellenmetadaten und Schemas, die sich aus dem Crawl ergeben.
- Schreiben Sie ein Python-ETL-Skript zum Extrahieren, Übertragen und Laden, das die Metadaten im Data Catalog für die folgenden Aufgaben verwendet:
  - Verknüpfen Sie die Daten in den verschiedenen Quelldateien in einer einzigen Datentabelle (d. h., denormalisieren Sie die Daten).
  - Filtern Sie die verknüpfte Tabelle in separate Tabellen nach Art des Gesetzgebers.
  - Schreiben Sie die resultierenden Daten in getrennte Apache-Parquet-Dateien, um sie später zu analysieren.

Die bevorzugte Methode zum Debuggen von Python oder PySpark Skripten während der Ausführung AWS ist die Verwendung von Notebooks auf AWS Glue Studio.

Schritt 1: Crawlen der Daten im Amazon S3 Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Folgen Sie den Schritten unterKonfiguration eines Crawlers, und erstellen Sie einen neuen Crawler, der den `s3://awsglue-datasets/examples/us-legislators/all` Datensatz in eine Datenbank crawlen kann, die `legislators` im AWS Glue-Datenkatalog benannt ist. Die Beispieldaten befinden sich bereits in diesem öffentlichen Amazon S3 Bucket.
3. Führen Sie den neuen Crawler aus und überprüfen Sie die `legislators`-Datenbank.

   Der Crawler erstellt die folgenden Metadatentabellen:

   - `persons_json`
   - `memberships_json`
   - `organizations_json`
   - `events_json`
   - `areas_json`
   - `countries_r_json`

   Dies ist eine teilweise normalisierte Sammlung von Tabellen mit Gesetzgebern und deren Geschichte.

Schritt 2: Hinzufügen des Boilerplate-Skripts zum Entwicklungsendpunkt-Notebook

Fügen Sie das folgende Standardskript in das Entwicklungsendpunkt-Notizbuch ein, um Folgendes zu importieren AWS Glue Bibliotheken, die Sie benötigen, und richten Sie eine einzelne ein: `GlueContext`

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Schritt 3: Untersuchen der Schemas aus den Daten im Data Catalog

Als Nächstes können Sie ganz einfach eine Datei DynamicFrame aus dem AWS Glue-Datenkatalog erstellen und die Schemas der Daten untersuchen. Wenn Sie beispielsweise das Schema der `persons_json`-Tabelle sehen möchten, fügen Sie Ihrem Notebook Folgendes hinzu:

```
persons = glueContext.create_dynamic_frame.from_catalog(
             database="legislators",
             table_name="persons_json")
print "Count: ", persons.count()
persons.printSchema()
```

Hier sehen Sie die Ausgabe des Druckaufrufs:

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
```

```
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Jede Person in der Tabelle ist Mitglied eines Kongressausschusses der USA.

Geben Sie zum Anzeigen des Schemas der `memberships_json`-Tabelle Folgendes ein:

```
memberships = glueContext.create_dynamic_frame.from_catalog(
                 database="legislators",
                 table_name="memberships_json")
print "Count: ", memberships.count()
memberships.printSchema()
```

Die Ausgabe sieht wie folgt aus:

```
Count:  10439
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string
```

Die `organizations` sind Parteien und die beiden Kammern des Kongresses, der Senat und das Repräsentantenhaus. Geben Sie zum Anzeigen des Schemas der `organizations_json`-Tabelle Folgendes ein:

```
orgs = glueContext.create_dynamic_frame.from_catalog(
           database="legislators",
           table_name="organizations_json")
print "Count: ", orgs.count()
orgs.printSchema()
```

Die Ausgabe sieht wie folgt aus:

```
Count:  13
root
|-- classification: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- name: string
|-- seats: int
|-- type: string
```

Schritt 4: Filtern der Daten

Als Nächstes behalten Sie nur die gewünschten Felder bei und benennen `id` in `org_id` um. Das Dataset ist klein genug, sodass Sie alles im Überblick ansehen können.

`toDF()` konvertiert einen `DynamicFrame` in einen Apache Spark `DataFrame`, sodass Sie die Transformationen, die bereits in Apache Spark SQL vorhanden sind, anwenden können:

```
orgs = orgs.drop_fields(['other_names',
                         'identifiers']).rename_field(
                             'id', 'org_id').rename_field(
                                'name', 'org_name')
orgs.toDF().show()
```

Nachfolgend sehen Sie die Ausgabe:

```
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|classification|              org_id|            org_name|               links|seats|
     type|               image|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|         party|            party/al|                  AL|                null| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/democrat-li...|    Democrat-Liberal|[[website,http://...| null|
     null|                null|
|   legislature|d56acebe-8fdc-47b...|House of Represen...|                null|  435|
lower house|                null|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|party/new_progres...|     New Progressive|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/popular_dem...|    Popular Democrat|[[website,http://...| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/republican-...|Republican-Conser...|[[website,http://...| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|   legislature|8fa6c3d2-71dc-478...|             Senate|                null|  100|
upper house|                null|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
```

Geben Sie Folgendes ein, um die `organizations` anzuzeigen, die in `memberships` erscheinen:

```
memberships.select_fields(['organization_id']).toDF().distinct().show()
```

Nachfolgend sehen Sie die Ausgabe:

```
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Schritt 5: Zusammenfügen aller Einzelteile

Verwenden Sie jetzt AWS Glue um diese relationalen Tabellen zu verbinden und eine vollständige Verlaufstabelle mit Gesetzgebern `memberships` und den entsprechenden Daten zu erstellen. `organizations`

1. Verknüpfen Sie zuerst `persons` und `memberships` mit `id` und `person_id`.
2. Verknüpfen Sie dann das Ergebnis mit `orgs` für `org_id` und `organization_id`.
3. Danach löschen Sie die redundanten Felder `person_id` und `org_id`.

Sie können alle diese Operationen in einer (erweiterten) Codezeile ausführen:

```
l_history = Join.apply(orgs,
                       Join.apply(persons, memberships, 'id', 'person_id'),
                       'org_id', 'organization_id').drop_fields(['person_id',
 'org_id'])
print "Count: ", l_history.count()
l_history.printSchema()
```

Die Ausgabe sieht wie folgt aus:

```
Count:  10439
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
```

```
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- death_date: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- family_name: string
|-- id: string
|-- start_date: string
|-- end_date: string
```

Sie haben jetzt die endgültige Tabelle vorliegen, die Sie für die Analyse verwenden können. Sie können es in einem kompakten, effizienten Analyseformat — nämlich Parquet — ausschreiben, in dem Sie SQL ausführen können AWS Glue, Amazon Athena oder Amazon Redshift Spectrum.

Der folgende Aufruf schreibt die Tabelle über mehrere Dateien, um schnelle parallele Lesevorgänge zu unterstützen, wenn die Analysen später ausgeführt werden:

```
glueContext.write_dynamic_frame.from_options(frame = l_history,
          connection_type = "s3",
```

```
          connection_options = {"path": "s3://glue-sample-target/output-dir/
legislator_history"},
          format = "parquet")
```

Um alle Historiendaten in eine einzige Datei zu schreiben, müssen Sie sie in ein Datenframe konvertieren, sie neu partitionieren und ausgeben:

```
s_history = l_history.toDF().repartition(1)
s_history.write.parquet('s3://glue-sample-target/output-dir/legislator_single')
```

Oder wenn Sie sie nach Senat und Repräsentantenhaus trennen möchten:

```
l_history.toDF().write.parquet('s3://glue-sample-target/output-dir/legislator_part',
                               partitionBy=['org_name'])
```

Schritt 6: Transformieren der Daten für relationale Datenbanken

AWS Glue macht es einfach, die Daten in relationale Datenbanken wie Amazon Redshift zu schreiben, selbst bei halbstrukturierten Daten. Der Service bietet eine Transformation für `relationalize`, die `DynamicFrames` vereinfacht, unabhängig von der Komplexität der Objekte im Frame.

Mit dem `l_history` `DynamicFrame` in diesem Beispiel übergeben Sie den Namen einer Stammtabelle (`hist_root`) und einen temporären Arbeitspfad an `relationalize`. Damit wird eine `DynamicFrameCollection` zurückgegeben. Anschließend können Sie die Namen der `DynamicFrames` in dieser Sammlung auflisten:

```
dfc = l_history.relationalize("hist_root", "s3://glue-sample-target/temp-dir/")
dfc.keys()
```

Das folgende Beispiel zeigt die Ausgabe des `keys`-Aufrufs:

```
[u'hist_root', u'hist_root_contact_details', u'hist_root_links',
 u'hist_root_other_names', u'hist_root_images', u'hist_root_identifiers']
```

Mit `Relationalize` wurde die Historientabelle in sechs neue Tabellen unterteilt: eine Stammtabelle mit einem Datensatz für jedes Objekt im `DynamicFrame`und Hilfstabellen für die Arrays. Die Array-Verarbeitung in relationalen Datenbanken ist oft suboptimal, vor allem, wenn diese Arrays sehr groß sind. Die Aufteilung der Arrays auf verschiedene Tabellen beschleunigt die Abfragen deutlich.

Sehen Sie sich dann die Aufteilung an, indem Sie `contact_details` untersuchen:

```
l_history.select_fields('contact_details').printSchema()
dfc.select('hist_root_contact_details').toDF().where("id = 10 or id =
 75").orderBy(['id','index']).show()
```

Das folgende Beispiel zeigt die Ausgabe des `show`-Aufrufs:

```
root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
| 10|    0|                    fax|                         |
| 10|    1|                       |             202-225-1314|
| 10|    2|                  phone|                         |
| 10|    3|                       |             202-225-3772|
| 10|    4|                twitter|                         |
| 10|    5|                       |          MikeRossUpdates|
| 75|    0|                    fax|                         |
| 75|    1|                       |             202-225-7856|
| 75|    2|                  phone|                         |
| 75|    3|                       |             202-225-2711|
| 75|    4|                twitter|                         |
| 75|    5|                       |                SenCapito|
+---+-----+-----------------------+-------------------------+
```

Das `contact_details`-Feld war ein Array von Strukturen im ursprünglichen `DynamicFrame`. Jedes Element dieser Arrays ist eine separate Zeile in der Hilfstabelle, indiziert durch `index`. Die `id` ist ein Fremdschlüssel in der `hist_root`-Tabelle mit dem Schlüssel `contact_details`:

```
dfc.select('hist_root').toDF().where(
    "contact_details = 10 or contact_details = 75").select(
       ['id', 'given_name', 'family_name', 'contact_details']).show()
```

Im Folgenden wird die Ausgabe dargestellt:

```
+--------------------+----------+-----------+---------------+
|                  id|given_name|family_name|contact_details|
+--------------------+----------+-----------+---------------+
|f4fc30ee-7b42-432...|      Mike|       Ross|             10|
|e3c60f34-7d1b-4c0...|   Shelley|     Capito|             75|
+--------------------+----------+-----------+---------------+
```

Beachten Sie bei diesen Befehlen, dass `toDF()` und dann ein `where`-Ausdruck zum Filtern der Zeilen, die Sie sehen möchten, verwendet werden.

Durch Verknüpfen der `hist_root`-Tabelle mit den Hilfstabellen können Sie die folgenden Aufgaben ausführen:

- Laden von Daten in Datenbanken ohne Array-Support
- Abfragen jedes einzelnen Elements in einem Array mithilfe von SQL

Speichern Sie Ihre Amazon Redshift Redshift-Anmeldeinformationen sicher und greifen Sie darauf zu mit einem AWS Glue Verbindung. Weitere Informationen zum Herstellen Ihrer eigenen Verbindung finden Sie unter Herstellen einer Verbindung zu Daten.

Sie können jetzt Ihre Daten für eine Verbindung schreiben, indem Sie die `DynamicFrames` einzeln durchlaufen:

```
for df_name in dfc.keys():
  m_df = dfc.select(df_name)
  print "Writing to table: ", df_name
  glueContext.write_dynamic_frame.from_jdbc_conf(frame = m_df, connection settings
 here)
```

Ihre Verbindungseinstellungen unterscheiden sich je nach Typ der relationalen Datenbank:

- Anweisungen zum Schreiben in Amazon Redshift finden Sie unter the section called “Redshift-Verbindungen”.
- Weitere Datenbanken finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

Schlussfolgerung

Insgesamt AWS Glue ist sehr flexibel. Sie können in wenigen Codezeilen Ziele erreichen, für die normalerweise mehrere Tage erforderlich wären. Sie finden die gesamten source-to-target ETL-Skripte in der Python-Datei `join_and_relationalize.py` im AWS Glue Beispiele auf GitHub.

Codebeispiel: Datenvorbereitung mit ResolveChoice, Lambda und ApplyMapping

Der Datensatz, der in diesem Beispiel verwendet wird, besteht aus Zahlungsdaten des Medicare-Anbieters, die aus zwei Data.cms.gov-Datensätzen heruntergeladen wurden: „Zusammenfassung der Anbieter von Inpatient Prospective Payment System Provider for the Top 100 Diagnosis-Related Groups — 011" und „Stationäre Gebührendaten für das Geschäftsjahr 2011". FY2 Nach dem Download änderten wir die Daten derart, dass wir einige fehlerhafte Datensätze am Ende der Datei hinzufügten. Diese geänderte Datei ist in einem öffentlichen Amazon S3 Bucket unter `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv` gespeichert.

Den Quellcode für dieses Beispiel finden Sie in der Datei im `data_cleaning_and_lambda.py` AWS Glue GitHub Beispiel-Repository.

Die bevorzugte Methode zum Debuggen von Python oder PySpark Skripten während der Ausführung AWS ist die Verwendung von Notebooks auf AWS Glue Studio.

Schritt 1: Crawlen der Daten im Amazon S3 Bucket

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Erstellen Sie nach dem unter beschriebenen Prozess einen neuen Crawler, der die `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv` Datei crawlen und die resultierenden Metadaten in einer Datenbank platzieren kann, die `payments` im AWS Glue-Datenkatalog benannt ist. Konfiguration eines Crawlers
3. Führen Sie den neuen Crawler aus und überprüfen Sie die `payments`-Datenbank. Sie sollten feststellen, dass der Crawler eine Metadatentabelle mit Namen `medicare` in der Datenbank angelegt hat, nachdem er den Anfang der Datei gelesen hat, um ihr Format und Trennzeichen zu bestimmen.

Das Schema der neuen `medicare`-Tabelle sieht wie folgt aus:

```
Column  name                              Data type
================================================
drg definition                              string
provider id                                 bigint
provider name                               string
provider street address                     string
provider city                               string
provider state                              string
provider zip code                           bigint
hospital referral region description        string
total discharges                            bigint
average covered charges                     string
average total payments                      string
average medicare payments                   string
```

Schritt 2: Hinzufügen des Boilerplate-Skripts zum Entwicklungsendpunkt-Notebook

Fügen Sie das folgende Standardskript in das Entwicklungsendpunkt-Notizbuch ein, um Folgendes zu importieren AWS Glue Bibliotheken, die Sie benötigen, und richten Sie eine einzelne ein: `GlueContext`

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Schritt 3: Vergleichen der verschiedene Schema Parsings

Als Nächstes können Sie sehen, ob das Schema, das von einem Apache Spark erkannt wurde, mit dem Schema identisch `DataFrame` ist, das Sie AWS Glue Crawler aufgezeichnet. Führen Sie diesen Code aus:

```

medicare = spark.read.format(
```

```
   "com.databricks.spark.csv").option(
   "header", "true").option(
   "inferSchema", "true").load(
   's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
medicare.printSchema()
```

Hier ist die Ausgabe des `printSchema`-Aufrufs:

```
root
 |-- DRG Definition: string (nullable = true)
 |-- Provider Id: string (nullable = true)
 |-- Provider Name: string (nullable = true)
 |-- Provider Street Address: string (nullable = true)
 |-- Provider City: string (nullable = true)
 |-- Provider State: string (nullable = true)
 |-- Provider Zip Code: integer (nullable = true)
 |-- Hospital Referral Region Description: string (nullable = true)
 |--  Total Discharges : integer (nullable = true)
 |--  Average Covered Charges : string (nullable = true)
 |--  Average Total Payments : string (nullable = true)
 |-- Average Medicare Payments: string (nullable = true)
```

Schauen Sie sich als Nächstes das Schema an, das ein AWS Glue `DynamicFrame`generiert:

```
medicare_dynamicframe = glueContext.create_dynamic_frame.from_catalog(
       database = "payments",
       table_name = "medicare")
medicare_dynamicframe.printSchema()
```

Die Ausgabe von `printSchema` sieht wie folgt aus:

```
root
 |-- drg definition: string
 |-- provider id: choice
 |    |-- long
 |    |-- string
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
```

```
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

Das `DynamicFrame` generiert ein Schema, in dem `provider id` entweder ein `long` oder ein `string` Typ sein kann. Das `DataFrame`-Schema listet `Provider Id` als `string`-Typ auf und der Data Catalog listet `provider id` als `bigint`-Typ auf.

Welches ist richtig? Es gibt zwei Datensätze am Ende der Datei (von 160.000 Datensätzen) mit `string`-Werten in dieser Spalte. Dies sind die fehlerhaften Datensätze, die eingefügt wurden, um ein Problem zu veranschaulichen.

Um diese Art von Problem zu lösen, AWS Glue `DynamicFrame`führt das Konzept eines Auswahltyps ein. In diesem Fall zeigt das `DynamicFrame`, dass sowohl `long` als auch `string` Werte in dieser Spalte erscheinen können. Das Tool AWS Glue Der Crawler hat die `string` Werte übersehen, weil er nur ein 2-MB-Präfix der Daten berücksichtigt hat. Der Apache Spark `DataFrame` berücksichtigt den gesamten Datensatz. Er wurde jedoch gezwungen, der Spalte den allgemeinsten Typ zuzuweisen (`string`). Tatsächlich greift Spark oft auf den allgemeinsten Fall zurück, wenn es komplexe Typen oder Variationen gibt, mit denen es nicht vertraut ist.

Um die `provider id`-Spalte abzufragen, lösen Sie zuerst den Auswahltyp auf. Sie können die `resolveChoice`-Transformationsmethode in Ihrem `DynamicFrame` verwenden, um diese `string` Werte in `long` Werte mit einer `cast:long` Option umzuwandeln:

```
medicare_res = medicare_dynamicframe.resolveChoice(specs = [('provider
 id','cast:long')])
medicare_res.printSchema()
```

Die `printSchema`-Ausgabe ist jetzt:

```
root
 |-- drg definition: string
 |-- provider id: long
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
```

```
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

Wo der Wert a war`string`, der nicht umgewandelt werden konnte, AWS Glue hat ein eingefügt`null`.

Eine weitere Möglichkeit besteht darin, den Auswahltyp in ein `struct` umzuwandeln, das die Werte beider Typen beibehält.

Als nächstes schauen Sie sich die Zeilen an, die nicht normal waren:

```
medicare_res.toDF().where("'provider id' is NULL").show()
```

Sie sehen folgendes:

```
+--------------------+----------+--------------+----------------------+-------------
+-------------+----------------+-------------------------------------+---------------
+----------------------+---------------------+------------------------+
|      drg definition|provider id|  provider name|provider street address|provider
 city|provider state|provider zip code|hospital referral region description|total
 discharges|average covered charges|average total payments|average medicare payments|
+--------------------+----------+--------------+----------------------+-------------
+-------------+----------------+-------------------------------------+---------------
+----------------------+---------------------+------------------------+
|948 - SIGNS & SYM...|       null|            INC|       1050 DIVISION ST|
 MAUSTON|            WI|            53948|                        WI - Madison|
      12|             $11961.41|             $4619.00|                $3775.33|
|948 - SIGNS & SYM...|       null| INC- ST JOSEPH|     5000 W CHAMBERS ST|
 MILWAUKEE|            WI|            53210|                        WI - Milwaukee|
        14|             $10514.28|             $5562.50|                $4522.78|
+--------------------+----------+--------------+----------------------+-------------
+-------------+----------------+-------------------------------------+---------------
+----------------------+---------------------+------------------------+
```

Entfernen Sie nun die beiden fehlerhaften Datensätze wie folgt:

```
medicare_dataframe = medicare_res.toDF()
```

```
medicare_dataframe = medicare_dataframe.where("'provider id' is NOT NULL")
```

Schritt 4: Zuordnen der Daten und Nutzung der Apache Spark Lambda Funktionen

AWS Glue unterstützt Lambda-Funktionen, auch bekannt als benutzerdefinierte Funktionen, noch nicht direkt. Aber Sie können jederzeit ein `DynamicFrame` zu und von Apache Spark `DataFrame` konvertieren, um die Vorteile der Spark-Funktionalität zusätzlich zu den speziellen Features von `DynamicFrames` zu nutzen.

Als nächstes wandeln Sie die Zahlungsinformationen in Zahlen um, sodass Analytik-Engines wie Amazon Redshift oder Amazon Athena ihre Zahlen schneller verarbeiten können:

```
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType

chop_f = udf(lambda x: x[1:], StringType())
medicare_dataframe = medicare_dataframe.withColumn(
        "ACC", chop_f(
            medicare_dataframe["average covered charges"])).withColumn(
                "ATP", chop_f(
                    medicare_dataframe["average total payments"])).withColumn(
                        "AMP", chop_f(
                            medicare_dataframe["average medicare payments"]))
medicare_dataframe.select(['ACC', 'ATP', 'AMP']).show()
```

Die Ausgabe des `show`-Aufrufs sieht wie folgt aus:

```
+--------+-------+-------+
|     ACC|    ATP|    AMP|
+--------+-------+-------+
|32963.07|5777.24|4763.73|
|15131.85|5787.57|4976.71|
|37560.37|5434.95|4453.79|
|13998.28|5417.56|4129.16|
|31633.27|5658.33|4851.44|
|16920.79|6653.80|5374.14|
|11977.13|5834.74|4761.41|
|35841.09|8031.12|5858.50|
|28523.39|6113.38|5228.40|
|75233.38|5541.05|4386.94|
|67327.92|5461.57|4493.57|
|39607.28|5356.28|4408.20|
```

```
|22862.23|5374.65|4186.02|
|31110.85|5366.23|4376.23|
|25411.33|5282.93|4383.73|
| 9234.51|5676.55|4509.11|
|15895.85|5930.11|3972.85|
|19721.16|6192.54|5179.38|
|10710.88|4968.00|3898.88|
|51343.75|5996.00|4962.45|
+--------+-------+-------+
only showing top 20 rows
```

Das sind alles Zeichenfolgen in den Daten. Wir können die leistungsstarke `apply_mapping`-Transformationsmethode verwenden, um die Daten zu löschen, umzubenennen, zu casten und zu verschachteln, sodass andere Datenprogrammiersprachen und -systeme leicht darauf zugreifen können:

```
from awsglue.dynamicframe import DynamicFrame
medicare_tmp_dyf = DynamicFrame.fromDF(medicare_dataframe, glueContext, "nested")
medicare_nest_dyf = medicare_tmp_dyf.apply_mapping([('drg definition', 'string', 'drg',
 'string'),
                ('provider id', 'long', 'provider.id', 'long'),
                ('provider name', 'string', 'provider.name', 'string'),
                ('provider city', 'string', 'provider.city', 'string'),
                ('provider state', 'string', 'provider.state', 'string'),
                ('provider zip code', 'long', 'provider.zip', 'long'),
                ('hospital referral region description', 'string','rr', 'string'),
                ('ACC', 'string', 'charges.covered', 'double'),
                ('ATP', 'string', 'charges.total_pay', 'double'),
                ('AMP', 'string', 'charges.medicare_pay', 'double')])
medicare_nest_dyf.printSchema()
```

Die `printSchema`-Ausgabe sieht wie folgt aus:

```
root
 |-- drg: string
 |-- provider: struct
 |    |-- id: long
 |    |-- name: string
 |    |-- city: string
 |    |-- state: string
 |    |-- zip: long
 |-- rr: string
```

```
 |-- charges: struct
 |    |-- covered: double
 |    |-- total_pay: double
 |    |-- medicare_pay: double
```

Wenn Sie die Daten wieder in ein Spark `DataFrame` verwandeln, können Sie sehen, wie sie jetzt aussehen:

```
medicare_nest_dyf.toDF().show()
```

Die Ausgabe sieht wie folgt aus:

```
+--------------------+--------------------+--------------+--------------------+
|                 drg|            provider|            rr|             charges|
+--------------------+--------------------+--------------+--------------------+
|039 - EXTRACRANIA...|[10001,SOUTHEAST ...|    AL - Dothan|[32963.07,5777.24...|
|039 - EXTRACRANIA...|[10005,MARSHALL M...|AL - Birmingham|[15131.85,5787.57...|
|039 - EXTRACRANIA...|[10006,ELIZA COFF...|AL - Birmingham|[37560.37,5434.95...|
|039 - EXTRACRANIA...|[10011,ST VINCENT...|AL - Birmingham|[13998.28,5417.56...|
|039 - EXTRACRANIA...|[10016,SHELBY BAP...|AL - Birmingham|[31633.27,5658.33...|
|039 - EXTRACRANIA...|[10023,BAPTIST ME...|AL - Montgomery|[16920.79,6653.8,...|
|039 - EXTRACRANIA...|[10029,EAST ALABA...|AL - Birmingham|[11977.13,5834.74...|
|039 - EXTRACRANIA...|[10033,UNIVERSITY...|AL - Birmingham|[35841.09,8031.12...|
|039 - EXTRACRANIA...|[10039,HUNTSVILLE...|AL - Huntsville|[28523.39,6113.38...|
|039 - EXTRACRANIA...|[10040,GADSDEN RE...|AL - Birmingham|[75233.38,5541.05...|
|039 - EXTRACRANIA...|[10046,RIVERVIEW ...|AL - Birmingham|[67327.92,5461.57...|
|039 - EXTRACRANIA...|[10055,FLOWERS HO...|    AL - Dothan|[39607.28,5356.28...|
|039 - EXTRACRANIA...|[10056,ST VINCENT...|AL - Birmingham|[22862.23,5374.65...|
|039 - EXTRACRANIA...|[10078,NORTHEAST ...|AL - Birmingham|[31110.85,5366.23...|
|039 - EXTRACRANIA...|[10083,SOUTH BALD...|    AL - Mobile|[25411.33,5282.93...|
|039 - EXTRACRANIA...|[10085,DECATUR GE...|AL - Huntsville|[9234.51,5676.55,...|
|039 - EXTRACRANIA...|[10090,PROVIDENCE...|    AL - Mobile|[15895.85,5930.11...|
|039 - EXTRACRANIA...|[10092,D C H REGI...|AL - Tuscaloosa|[19721.16,6192.54...|
|039 - EXTRACRANIA...|[10100,THOMAS HOS...|    AL - Mobile|[10710.88,4968.0,...|
|039 - EXTRACRANIA...|[10103,BAPTIST ME...|AL - Birmingham|[51343.75,5996.0,...|
+--------------------+--------------------+--------------+--------------------+
only showing top 20 rows
```

Schritt 5: Schreiben der Daten in Apache Parquet

AWS Glue macht es einfach, die Daten in einem Format wie Apache Parquet zu schreiben, das relationale Datenbanken effektiv nutzen können:

```
glueContext.write_dynamic_frame.from_options(
       frame = medicare_nest_dyf,
       connection_type = "s3",
       connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
       format = "parquet")
```

AWS Referenz für PySpark Klebererweiterungen

AWS Glue hat die folgenden Erweiterungen für den PySpark Python-Dialekt erstellt.

- Zugriff auf Parameter mit `getResolvedOptions`
- PySpark Erweiterungstypen
- DynamicFrame Klasse
- DynamicFrameCollection Klasse
- DynamicFrameWriter Klasse
- DynamicFrameReader Klasse
- GlueContext Klasse

Zugriff auf Parameter mit **`getResolvedOptions`**

Das Tool AWS Glue `getResolvedOptions(args, options)`Die Hilfsfunktion gibt Ihnen Zugriff auf die Argumente, die an Ihr Skript übergeben werden, wenn Sie einen Job ausführen. Um diese Funktion zu verwenden, importieren Sie sie zunächst aus AWS Glue `utils`Modul, zusammen mit dem `sys` Modul:

```
import sys
from awsglue.utils import getResolvedOptions
```

`getResolvedOptions(args, options)`

- `args` – Die Liste der Argumente in `sys.argv`.
- `options` – Ein Python-Array der Argumentnamen, die Sie abrufen möchten.

Example Abrufen von Argumenten, die an a übergeben wurden JobRun

Angenommen, Sie haben JobRun in einem Skript, vielleicht in einer Lambda-Funktion, eine erstellt:

```
response = client.start_job_run(
             JobName = 'my_test_Job',
             Arguments = {
               '--day_partition_key':   'partition_0',
               '--hour_partition_key':  'partition_1',
               '--day_partition_value':  day_partition_value,
               '--hour_partition_value': hour_partition_value } )
```

Zum Abrufen der Argumente, die übergeben werden, können Sie die `getResolvedOptions`-Funktion wie folgt verwenden:

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
                           'hour_partition_key',
                           'day_partition_value',
                           'hour_partition_value'])
print "The day-partition key is: ", args['day_partition_key']
print "and the day-partition value is: ", args['day_partition_value']
```

Beachten Sie, dass die einzelnen Argumente zwar jeweils als mit zwei Bindestrichen beginnend definiert werden, im Skript jedoch ohne Bindestriche referenziert werden. Die Argumente verwenden nur Unterstriche, keine Bindestriche. Ihre Argumente müssen dieser Konvention folgen, um aufgelöst werden zu können.

PySpark Erweiterungstypen

Die Typen, die von der verwendet werden AWS Glue PySpark Erweiterungen.

DataType

Die Basisklasse für die anderen AWS Glue-Typen.

`__init__(properties={})`

- `properties` – Eigenschaften des Datentyps (optional).

`typeName(cls)`

Gibt den Typ des AWS Glue Typklasse (d. h. der Klassenname, bei dem „Type“ am Ende entfernt wurde).

- `cls`— Ein AWS Glue Klasseninstanz, abgeleitet von`DataType`.

`jsonValue( )`

Gibt ein JSON-Objekt zurück, das den Datentyp und die Eigenschaften der Klasse enthält:

```
{
  "dataType": typeName,
  "properties": properties
}
```

AtomicType und einfache Ableitungen

Erbt von der DataType Klasse und erweitert sie und dient als Basisklasse für alle AWS Glue atomare Datentypen.

`fromJsonValue(cls, json_value)`

Initialisiert eine Klassen-Instance mit Werten aus einem JSON-Objekt.

- `cls`— Ein AWS Glue Geben Sie die zu initialisierende Klasseninstanz ein.
- `json_value` – Das JSON-Objekt, von dem Schlüssel-Wert-Paare geladen werden sollen.

Die folgenden Typen sind einfache Derivate der Klasse AtomicType:

- `BinaryType` – Binäre Daten.
- `BooleanType` – Boolesche Werte.
- `ByteType` – Ein Byte-Wert.
- `DateType` – Ein Datums-/Uhrzeitwert.
- `DoubleType` – Ein Gleitkommawert mit doppelter Präzision.
- `IntegerType` – Ein Ganzzahlwert.

- `LongType` – Eine lange Ganzzahl.
- `NullType` – Ein Null-Wert.
- `ShortType` – Eine kurze Ganzzahl.
- `StringType` – Eine Textzeichenfolge.
- `TimestampType` – Ein Zeitstempelwert (in der Regel in Sekunden von 1.1.1970).
- `UnknownType` – Ein Wert nicht identifizierten Typs.

DecimalType(AtomicType)

Übernahme und Erweiterung der Klasse AtomicType, um eine Dezimalzahl darzustellen (eine Zahl ausgedrückt in Dezimalstellen anstelle von binären Basis-2-Zahlen).

`__init__(precision=10, scale=2, properties={})`

- `precision` – Die Anzahl der Ziffern in der Dezimalzahl (optional; Standardwert ist 10).
- `scale` – Die Anzahl der Ziffern rechts vom Dezimaltrennzeichen (optional; Standardwert ist 2).
- `properties` – Die Eigenschaften der Dezimalzahl (optional).

EnumType(AtomicType)

Übernahme und Erweiterung der Klasse AtomicType, um eine Aufzählung gültiger Optionen darzustellen.

`__init__(options)`

- `options` – Eine Liste der aufzuzählenden Optionen.

Sammlungstypen

- ArrayType(DataType)
- ChoiceType(DataType)
- MapType(DataType)
- Field(Object)
- StructType(DataType)
- EntityType(DataType)

ArrayType(DataType)

`__init__(elementType=UnknownType(), properties={})`

- `elementType`— Der Typ der Elemente im Array (optional; die Standardeinstellung ist UnknownType).
- `properties` – Eigenschaften des Arrays (optional).

ChoiceType(DataType)

`__init__(choices=[], properties={})`

- `choices` – Eine Liste möglicher Optionen (optional).
- `properties` – Eigenschaften dieser Optionen (optional).

`add(new_choice)`

Fügt eine neue Option zur Liste der möglichen Optionen hinzu.

- `new_choice` – Die Option, die der Liste der möglichen Optionen hinzugefügt werden soll.

`merge(new_choices)`

Führt eine Liste neuer Optionen mit der vorhandenen Auswahlliste zusammen.

- `new_choices` – Eine Liste neuer Optionen, die mit vorhandenen Optionen zusammengeführt werden sollen.

MapType(DataType)

`__init__(valueType=UnknownType, properties={})`

- `valueType`— Der Typ der Werte in der Map (optional; die Standardeinstellung ist UnknownType).
- `properties` – Eigenschaften der Zuordnung (optional).

Field(Object)

Erstellt ein Feldobjekt aus einem Objekt, das von DataType abgeleitet wird.

`__init__(name, dataType, properties={})`

- `name` – Der Name, der dem Feld zugewiesen werden soll.
- `dataType` – Das Objekt, von dem ein Feld erstellt werden soll.
- `properties` – Eigenschaften des Felds (optional).

StructType(DataType)

Definiert eine Datenstruktur (`struct`).

`__init__(fields=[], properties={})`

- `fields` – Eine Liste der Felder (des Typs `Field`), die in der Struktur enthalten sein sollen (optional).
- `properties` – Eigenschaften der Struktur (optional).

`add(field)`

- `field` – Ein Objekt vom Typ `Field`, das der Struktur hinzugefügt werden soll.

`hasField(field)`

Gibt `True` zurück, wenn diese Struktur über ein Feld mit demselben Namen verfügt, andernfalls `False`.

- `field` – Ein Feldname oder ein Objekt vom Typ `Field`, dessen Name verwendet wird.

`getField(field)`

- `field` – Ein Feldname oder ein Objekt vom Typ `Field`, dessen Name verwendet wird. Wenn die Struktur über ein Feld mit demselben Namen verfügt, wird er zurückgegeben.

EntityType(DataType)

`__init__(entity, base_type, properties)`

Diese Klasse ist noch nicht implementiert.

andere Typen

- DataSource(Objekt)
- DataSink(Objekt)

DataSource(Objekt)

`__init__(j_source, sql_ctx, name)`

- `j_source` – Die Datenquelle.
- `sql_ctx` – Der SQL-Kontext.
- `name` – Der Datenquellenname.

`setFormat(format, **options)`

- `format` – Das Format, das für die Datenquelle festgelegt werden soll.
- `options` – Eine Sammlung von Optionen für die Datenquelle. Weitere Informationen zu den Formatoptionen finden Sie unter the section called “Pfad-Formatoptionen”.

`getFrame()`

Gibt ein `DynamicFrame` für die Datenquelle zurück.

DataSink(Objekt)

`__init__(j_sink, sql_ctx)`

- `j_sink` – Zu erstellende Senke.
- `sql_ctx` – Der SQL-Kontext für die Datensenke.

`setFormat(format, **options)`

- `format` – Das Format, das für die Datensenke festgelegt werden soll.
- `options` – Eine Sammlung von Optionen für die Datensenke. Weitere Informationen zu den Formatoptionen finden Sie unter the section called “Pfad-Formatoptionen”.

`setAccumulableSize(size)`

- `size` – Die festzulegende akkumulierte Größe, in Byte.

`writeFrame(dynamic_frame, info="")`

- `dynamic_frame` – Der zu schreibende `DynamicFrame`.
- `info` – Informationen zum `DynamicFrame` (optional).

`write(dynamic_frame_or_dfc, info="")`

Schreibt einen `DynamicFrame` oder eine `DynamicFrameCollection`.

- `dynamic_frame_or_dfc` – Ein zu schreibendes `DynamicFrame`-Objekt oder `DynamicFrameCollection`-Objekt.
- `info` – Informationen zum `DynamicFrame` oder zur `DynamicFrames`, der/die geschrieben werden soll (optional).

DynamicFrame Klasse

Einer der größten Abstraktionen in Apache Spark ist der SparkSQL `DataFrame`, der mit der `DataFrame`-Erstellung in R und Pandas vergleichbar ist. A `DataFrame` ähnelt einer Tabelle und unterstützt Operationen im Funktionsstil (map/reduce/filter/etc.) und SQL-Operationen (Select, Project, Aggregate).

`DataFrames` sind leistungsstark und werden umfassend genutzt, aber sie haben Einschränkungen in Bezug auf ETL (Extrahieren, Transformieren, Laden)-Operationen. Vor allem benötigen sie ein Schema, das angegeben werden muss, bevor Daten geladen werden. SparkSQL löst dies, indem es zwei Durchgänge für die Daten ausführt – den ersten für die Ableitung des Schemas und den zweiten

für das Laden der Daten. Diese Ableitung ist jedoch begrenzt und geht nicht die Gegebenheiten unübersichtlicher Daten an. Beispielsweise kann dasselbe Feld in verschiedenen Datensätzen unterschiedliche Typen aufweisen. Apache Spark gibt oft auf und meldet den Typ mithilfe des ursprünglichen Feldtexts als `string`. Dies ist möglicherweise nicht korrekt, und Sie wünschen sich eine bessere Kontrolle darüber, wie Schemaabweichungen gelöst werden. Und ein zusätzlicher Durchgang für die Quelldaten kann für große Datensätze unerschwinglich teuer sein.

Um diese Einschränkungen zu beheben, führt AWS Glue die ein`DynamicFrame`. Ein `DynamicFrame` ähnelt einem `DataFrame` mit der Ausnahme, dass jeder Datensatz selbstbeschreibend ist, sodass zunächst kein Schema erforderlich ist. Stattdessen AWS Glue berechnet on-the-fly bei Bedarf ein Schema und codiert Schemainkonsistenzen explizit mithilfe eines Auswahltyps (oder Unionstyps). Sie können diese Inkonsistenzen lösen, um Ihre Datensätze mit Datenspeichern kompatibel zu machen, die ein festes Schema erfordern.

Gleichermaßen stellt ein `DynamicRecord` einen logischen Datensatz innerhalb einem `DynamicFrame` dar. Dies ist mit einer Zeile in einem Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen. Wenn Sie AWS Glue mit verwenden PySpark, manipulieren Sie normalerweise nicht unabhängig`DynamicRecords`. Stattdessen transformieren Sie den Datensatz gemeinsam über seinen `DynamicFrame`.

Sie können `DynamicFrames` zu und aus `DataFrames` konvertieren, sobald Sie alle Schemainkonsistenzen gelöst haben.

– Konstruktion –

- __init__
- fromDF
- toDF

__init__

`__init__(jdf, glue_ctx, name)`

- `jdf` – Ein Verweis auf den Daten-Frame in der Java Virtual Machine (JVM).
- `glue_ctx` – Ein GlueContext Klasse-Objekt.
- `name` – Eine optionale Namenszeichenfolge, standardmäßig leer.

fromDF

`fromDF(dataframe, glue_ctx, name)`

Wandelt einen `DataFrame` in einen `DynamicFrame` um, indem `DataFrame`-Felder in `DynamicRecord`-Felder konvertiert werden. Gibt den neuen `DynamicFrame` zurück.

Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.

Diese Funktion geht davon aus, dass Spalten mit doppeltem Namen in Ihrem `DataFrame` bereits umbenannt wurden.

- `dataframe` – Der umzuwandelnde Apache Spark SQL `DataFrame` (erforderlich).
- `glue_ctx` – Das GlueContext Klasse-Objekt, das den Kontext für diese Transformation angibt (erforderlich).
- `name`— Der Name des Ergebnisses `DynamicFrame` (optional seit AWS Glue 3.0).

toDF

`toDF(options)`

Wandelt einen `DynamicFrame` in einen Apache Spark `DataFrame` um, indem `DynamicRecords` in `DataFrame`-Felder konvertiert werden. Gibt den neuen `DataFrame` zurück.

Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.

- `options` – Eine Liste der Optionen. Ermöglicht es Ihnen, zusätzliche Optionen für den Konvertierungsprozess anzugeben. Einige gültige Optionen, die Sie mit dem Parameter `options` verwenden können:
  - `format`— gibt das Format der Daten an, z. B. json, csv, parquet).
  - `separater or sep`— gibt für CSV-Dateien das Trennzeichen an.
  - `header`— gibt bei CSV-Dateien an, ob es sich bei der ersten Zeile um eine Kopfzeile handelt (wahr/falsch).
  - `inferSchema`— weist Spark an, das Schema automatisch abzuleiten (true/false).

Hier ist ein Beispiel für die Verwendung des Parameters `options` mit der `ToDF`-Methode:

```
from awsglue.context import GlueContext
from awsglue.dynamicframe import DynamicFrame
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

csv_dyf = glueContext.create_dynamic_frame.from_options(
  connection_type="s3",
  connection_options={"paths": ["s3://my-bucket/path/to/csv/"]},
  format="csv"
)
csv_cf = csv_dyf.toDF(options={
  "separator": ",",
  "header": "true",
  "inferSchema": "true"
})
```

Geben Sie den Zieltyp an, wenn Sie die Aktionstypen `Project` und `Cast` auswählen. Beispiele sind unter anderem:

```
>>>toDF([ResolveOption("a.b.c", "KeepAsStruct")])
>>>toDF([ResolveOption("a.b.c", "Project", DoubleType())])
```

– Informationen –

- count
- schema
- printSchema
- show
- repartition
- COALESCE

count

`count( )` – Gibt die Anzahl von Zeilen in dem zugrunde liegenden `DataFrame` zurück.

schema

`schema( )` – Gibt das Schema dieses `DynamicFrame` zurück oder wenn dieser nicht verfügbar ist, das Schema des zugrunde liegenden `DataFrame`.

Weitere Informationen über die `DynamicFrame`-Typen, aus denen dieses Schema besteht, finden Sie unter the section called “Typen”.

printSchema

`printSchema( )` – Druckt das Schema des zugrunde liegenden `DataFrame`.

show

`show(num_rows)` – Druckt eine bestimmte Anzahl von Zeilen aus dem zugrunde liegenden `DataFrame`.

repartition

`repartition(numPartitions)` – Gibt einen neuen `DynamicFrame` mit `numPartitions`-Partitionen zurück.

COALESCE

`coalesce(numPartitions)` – Gibt einen neuen `DynamicFrame` mit `numPartitions`-Partitionen zurück.

– Transformationen –

- apply_mapping
- drop_fields
- Filter
- join
- map
- mergeDynamicFrame
- relationalize
- rename_field

- resolveChoice
- select_fields
- spigot
- split_fields
- split_rows
- unbox
- the section called “union”
- unnest
- unnest_ddb_json
- write

apply_mapping

`apply_mapping(mappings, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Wendet ein deklaratives Mapping für einen `DynamicFrame` an und gibt einen neuen `DynamicFrame` mit diesen angewendeten Mappings auf die von Ihnen angegebenen Felder zurück. Nicht spezifizierte Felder werden in den neuen `DynamicFrame` weggelassen.

- `mappings` – Eine Liste von Mapping-Tupeln (erforderlich). Jede besteht aus (Quellspalte, Quelltyp, Zielspalte, Zieltyp).

  Wenn die Quellspalte einen Punkt “`.`„ im Namen hat, müssen Sie Backticks “`` ` ``„ darum herum platzieren. Um beispielsweise `this.old.name` (Zeichenfolge) auf `thisNewName` abzubilden, würden Sie das folgende Tupel verwenden:

  ```
  ("`this.old.name`", "string", "thisNewName", "string")
  ```

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie apply_mapping, um Felder umzubenennen und Feldtypen zu ändern

Im folgenden Beispielcode wird gezeigt, wie Sie die `apply_mapping`-Methode verwenden, um ausgewählte Felder umzubenennen und Feldtypen zu ändern.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use apply_mapping to reshape source data into
# the desired column names and types as a new DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Select and rename fields, change field type
print("Schema for the persons_mapped DynamicFrame, created with apply_mapping:")
persons_mapped = persons.apply_mapping(
    [
        ("family_name", "String", "last_name", "String"),
        ("name", "String", "first_name", "String"),
        ("birth_date", "String", "date_of_birth", "Date"),
    ]
```

```
)
persons_mapped.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_mapped DynamicFrame, created with apply_mapping:
root
|-- last_name: string
|-- first_name: string
|-- date_of_birth: date
```

drop_fields

`drop_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Ruft die FlatMap Klasse-Transformation auf, um Felder aus einem `DynamicFrame` zu entfernen. Gibt einen neuen `DynamicFrame` zurück, in dem die angegebenen Felder gelöscht wurden.

- `paths` – Eine Liste von Zeichenfolgen. Jeder enthält den vollständigen Pfad zu einem Feldknoten, den Sie löschen möchten. Sie können die Punktnotation verwenden, um verschachtelte Felder anzugeben. Wenn zum Beispiel Feld `first` ein untergeordnetes Feld des Feldes `name` im Baum ist, spezifizieren Sie `"name.first"` für den Pfad.

  Wenn der Name eines Feldknotens ein Literal `.` enthält, müssen Sie den Namen in Backticks (`` ` ``) einschließen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie drop_fields, um Felder aus einem **`DynamicFrame`** zu entfernen

Dieses Codebeispiel verwendet die `drop_fields`-Methode zum Entfernen ausgewählter Felder der obersten Ebene und verschachtelter Felder aus einem `DynamicFrame`.

Beispieldatensatz

Das Beispiel verwendet den folgenden Datensatz, der durch die `EXAMPLE-FRIENDS-DATA`-Tabelle im Code dargestellt wird:

```
{"name": "Sally", "age": 23, "location": {"state": "WY", "county": "Fremont"},
 "friends": []}
{"name": "Varun", "age": 34, "location": {"state": "NE", "county": "Douglas"},
 "friends": [{"name": "Arjun", "age": 3}]}
{"name": "George", "age": 52, "location": {"state": "NY"}, "friends": [{"name":
 "Fred"}, {"name": "Amy", "age": 15}]}
{"name": "Haruki", "age": 21, "location": {"state": "AK", "county": "Denali"}}
{"name": "Sheila", "age": 63, "friends": [{"name": "Nancy", "age": 22}]}
```

Beispiel-Code

```
# Example: Use drop_fields to remove top-level and nested fields from a DynamicFrame.
# Replace MY-EXAMPLE-DATABASE with your Glue Data Catalog database name.
# Replace EXAMPLE-FRIENDS-DATA with your table name.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame from Glue Data Catalog
glue_source_database = "MY-EXAMPLE-DATABASE"
glue_source_table = "EXAMPLE-FRIENDS-DATA"

friends = glueContext.create_dynamic_frame.from_catalog(
    database=glue_source_database, table_name=glue_source_table
)
print("Schema for friends DynamicFrame before calling drop_fields:")
friends.printSchema()

# Remove location.county, remove friends.age, remove age
friends = friends.drop_fields(paths=["age", "location.county", "friends.age"])
print("Schema for friends DynamicFrame after removing age, county, and friend age:")
friends.printSchema()
```

Output

```
Schema for friends DynamicFrame before calling drop_fields:
root
|-- name: string
```

```
|-- age: int
|-- location: struct
|    |-- state: string
|    |-- county: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
|    |    |-- age: int

Schema for friends DynamicFrame after removing age, county, and friend age:
root
|-- name: string
|-- location: struct
|    |-- state: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
```

Filter

`filter(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Gibt einen neuen `DynamicFrame` zurück, der alle `DynamicRecords` innerhalb des Eingabe-`DynamicFrame` enthält, die die angegebene Prädikat-Funktion `f` erfüllen.

- `f` – Die Prädikat-Funktion, die auf den `DynamicFrame` angewendet werden soll. Die Funktion muss einen `DynamicRecord` als Argument enthalten und "True" zurückgeben, wenn der `DynamicRecord` die Filteranforderungen erfüllt, oder andernfalls "False" (erforderlich).

  Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).

- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie Filter, um eine gefilterte Auswahl von Feldern zu erhalten

In diesem Beispiel wird verwendet die `filter`-Methode zum Erstellen eines neuen `DynamicFrame` verwendet, das eine gefilterte Auswahl eines anderen `DynamicFrame`-Feldes beinhaltet.

Wie die `map`-Methode nimmt `filter` eine Funktion als Argument, das auf jeden Datensatz im Original-`DynamicFrame`angewendet wird. Die Funktion nimmt einen Datensatz als Eingabe und gibt einen booleschen Wert zurück. Wenn der Rückgabewert true ist, wird der Datensatz in das Ergebnis von `DynamicFrame` aufgenommen. Wenn es false ist, wird der Datensatz weggelassen.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Datenvorbereitung mit ResolveChoice, Lambda und ApplyMapping und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use filter to create a new DynamicFrame
# with a filtered selection of records

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame from Glue Data Catalog
medicare = glueContext.create_dynamic_frame.from_options(
    "s3",
    {
        "paths": [
            "s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv"
```

```
        ]
    },
    "csv",
    {"withHeader": True},
)

# Create filtered DynamicFrame with custom lambda
# to filter records by Provider State and Provider City
sac_or_mon = medicare.filter(
    f=lambda x: x["Provider State"] in ["CA", "AL"]
    and x["Provider City"] in ["SACRAMENTO", "MONTGOMERY"]
)

# Compare record counts
print("Unfiltered record count: ", medicare.count())
print("Filtered record count:  ", sac_or_mon.count())
```

Output

```
Unfiltered record count:  163065
Filtered record count:   564
```

join

`join(paths1, paths2, frame2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Führt einen Equality Join mit einem anderen `DynamicFrame` durch und gibt den sich ergebenden `DynamicFrame` zurück.

- `paths1` – Eine Liste der Schlüssel in diesem Frame für einen Join.
- `paths2` – Eine Liste der Schlüssel in dem anderen Frame für einen Join.
- `frame2` – Der andere `DynamicFrame` für einen Join.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden von join zum Kombinieren von **`DynamicFrames`**

In diesem Beispiel wird die `join` Methode verwendet, um eine Verknüpfung von drei Objekten durchzuführen. `DynamicFrames` AWS Glue führt die Verbindung auf der Grundlage der von Ihnen angegebenen Feldtasten durch. Der daraus resultierende `DynamicFrame` enthält Zeilen aus den beiden Originalframes, in denen die angegebenen Schlüssel übereinstimmen.

Beachten Sie, dass `join` transform alle Felder intakt hält. Das bedeutet, dass die Felder, die Sie für den Abgleich angeben DynamicFrame, im Ergebnis angezeigt werden, auch wenn sie redundant sind und dieselben Schlüssel enthalten. In diesem Beispiel verwenden wir `drop_fields`, um diese redundanten Schlüssel nach dem Join zu entfernen.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use join to combine data from three DynamicFrames

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load DynamicFrames from Glue Data Catalog
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
memberships = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="memberships_json"
)
orgs = glueContext.create_dynamic_frame.from_catalog(
```

```
    database="legislators", table_name="organizations_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()
print("Schema for the memberships DynamicFrame:")
memberships.printSchema()
print("Schema for the orgs DynamicFrame:")
orgs.printSchema()

# Join persons and memberships by ID
persons_memberships = persons.join(
    paths1=["id"], paths2=["person_id"], frame2=memberships
)

# Rename and drop fields from orgs
# to prevent field name collisions with persons_memberships
orgs = (
    orgs.drop_fields(["other_names", "identifiers"])
    .rename_field("id", "org_id")
    .rename_field("name", "org_name")
)

# Create final join of all three DynamicFrames
legislators_combined = orgs.join(
    paths1=["org_id"], paths2=["organization_id"], frame2=persons_memberships
).drop_fields(["person_id", "org_id"])

# Inspect the schema for the joined data
print("Schema for the new legislators_combined DynamicFrame:")
legislators_combined.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
```

```
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the memberships DynamicFrame:
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string

Schema for the orgs DynamicFrame:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
```

```
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

Schema for the new legislators_combined DynamicFrame:
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
```

```
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

map

`map(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Gibt einen neuen `DynamicFrame` zurück, der sich durch Anwendung von Zuweisungsfunktionen auf alle Datensätze im ursprünglichen `DynamicFrame` ergibt.

- `f` – Die Zuweisungsfunktion, die auf alle Datensätze im `DynamicFrame` angewendet werden soll. Die Funktion muss einen `DynamicRecord` als Argument enthalten und gibt einen neuen `DynamicRecord` zurück (erforderlich).

  Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Apache Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die mit Fehlern in der Transformation im Zusammenhang steht (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Beispiel: Verwenden von map zur Anwendung einer Funktion auf jeden Datensatz in einem **`DynamicFrame`**

In diesem Beispiel wird gezeigt, wie Sie die `map`-Methode zum Anwenden einer Funktion auf jeden Datensatz eines `DynamicFrame` anwenden. Insbesondere wendet dieses Beispiel eine Funktion

namens `MergeAddress` auf jeden Datensatz an, um mehrere Adressfelder zu einem einzigen `struct`-Typ zusammenzuführen.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Datenvorbereitung mit ResolveChoice, Lambda und ApplyMapping und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use map to combine fields in all records
# of a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
medicare = glueContext.create_dynamic_frame.from_options(
        "s3",
        {"paths": ["s3://awsglue-datasets/examples/medicare/
Medicare_Hospital_Provider.csv"]},
        "csv",
        {"withHeader": True})
print("Schema for medicare DynamicFrame:")
medicare.printSchema()

# Define a function to supply to the map transform
# that merges address fields into a single field
def MergeAddress(rec):
  rec["Address"] = {}
  rec["Address"]["Street"] = rec["Provider Street Address"]
  rec["Address"]["City"] = rec["Provider City"]
  rec["Address"]["State"] = rec["Provider State"]
  rec["Address"]["Zip.Code"] = rec["Provider Zip Code"]
  rec["Address"]["Array"] = [rec["Provider Street Address"], rec["Provider City"],
 rec["Provider State"], rec["Provider Zip Code"]]
  del rec["Provider Street Address"]
  del rec["Provider City"]
```

```
  del rec["Provider State"]
  del rec["Provider Zip Code"]
  return rec


# Use map to apply MergeAddress to every record
mapped_medicare = medicare.map(f = MergeAddress)
print("Schema for mapped_medicare DynamicFrame:")
mapped_medicare.printSchema()
```

Output

```
Schema for medicare DynamicFrame:
root
|-- DRG Definition: string
|-- Provider Id: string
|-- Provider Name: string
|-- Provider Street Address: string
|-- Provider City: string
|-- Provider State: string
|-- Provider Zip Code: string
|-- Hospital Referral Region Description: string
|-- Total Discharges: string
|-- Average Covered Charges: string
|-- Average Total Payments: string
|-- Average Medicare Payments: string

Schema for mapped_medicare DynamicFrame:
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
```

```
|-- Provider Name: string
```

mergeDynamicFrame

`mergeDynamicFrame(stage_dynamic_frame, primary_keys, transformation_ctx = "", options = {}, info = "", stageThreshold = 0, totalThreshold = 0)`

Führt dieses `DynamicFrame` mit einem Staging-`DynamicFrame` basierend auf den angegebenen Primärschlüsseln zusammen, um Datensätze zu identifizieren. Doppelte Datensätze (Datensätze mit denselben Primärschlüsseln) werden nicht dedupliziert. Wenn kein übereinstimmender Datensatz im Staging-Frame vorhanden ist, werden alle Datensätze (einschließlich Duplikate) von der Quelle beibehalten. Wenn der Staging-Frame übereinstimmende Datensätze enthält, überschreiben die Datensätze aus dem Staging-Frame die Datensätze in der Quelle AWS Glue.

- `stage_dynamic_frame` – Der Staging-`DynamicFrame`, der zusammengeführt werden soll.
- `primary_keys` – Die Liste der Primärschlüsselfelder, die Datensätze aus den Quell- und dynamischen Staging-Frames abgleichen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge, die zum Abrufen von Metadaten über die aktuelle Transformation verwendet wird (optional).
- `options` – Eine Zeichenfolge von JSON-Name-Wert-Paaren, die zusätzliche Informationen für diese Transformation bereitstellen. Dieses Argument wird derzeit nicht verwendet.
- `info` – Ein `String`. Jede Zeichenfolge, die mit Fehlern in dieser Transformation verknüpft werden soll.
- `stageThreshold` – Ein `Long`. Die Anzahl der Fehler in der angegebenen Transformation, für die die Verarbeitung fehlerhaft sein muss.
- `totalThreshold` – Ein `Long`. Die Gesamtzahl der Fehler bis einschließlich dieser Transformation, bei denen die Verarbeitung fehlerhaft sein muss.

Diese Methode gibt einen neuen `DynamicFrame` zurück, der erhalten wird, indem Sie diesen `DynamicFrame` mit dem Staging-`DynamicFrame` zusammenführen.

Der zurückgegebene `DynamicFrame` enthält Datensatz A in folgenden Fällen:

- Wenn sowohl im Quell- als auch im Staging-Frame `A` vorhanden ist, wird `A` im Staging-Frame zurückgegeben.
- Wenn sich `A` in der Quelltabelle und `A.primaryKeys` nicht in `stagingDynamicFrame` befindet, wird `A` nicht in der Staging-Tabelle aktualisiert.

Der Quell- und der Staging-Frame müssen nicht dasselbe Schema haben.

Beispiel: Wird verwendet mergeDynamicFrame , um zwei auf der **`DynamicFrames`** Grundlage eines Primärschlüssels zusammenzuführen

Das folgende Codebeispiel zeigt, wie die `mergeDynamicFrame`-Methode verwendet wird, um einen `DynamicFrame` mit einem „Staging“-`DynamicFrame`, basierend auf dem Primärschlüssel `id`, zusammenzuführen.

Beispieldatensatz

Das Beispiel verwendet zwei `DynamicFrames` von einem `DynamicFrameCollection` mit dem Namen `split_rows_collection`. Die folgende Liste enthält die Schlüssel für `split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Beispiel-Code

```
# Example: Use mergeDynamicFrame to merge DynamicFrames
# based on a set of specified primary keys

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Inspect the original DynamicFrames
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
print("Inspect the DynamicFrame that contains rows where ID < 10")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
print("Inspect the DynamicFrame that contains rows where ID > 10")
frame_high.toDF().show()

# Merge the DynamicFrames based on the "id" primary key
merged_high_low = frame_high.mergeDynamicFrame(
    stage_dynamic_frame=frame_low, primary_keys=["id"]
)

# View the results where the ID is 1 or 20
print("Inspect the merged DynamicFrame that contains the combined rows")
```

```
merged_high_low.toDF().where("id = 1 or id= 20").orderBy("id").show()
```

Output

```
Inspect the DynamicFrame that contains rows where ID < 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                    fax|           202-225-3307|
|  1|    1|                  phone|           202-225-5731|
|  2|    0|                    fax|           202-225-3307|
|  2|    1|                  phone|           202-225-5731|
|  3|    0|                    fax|           202-225-3307|
|  3|    1|                  phone|           202-225-5731|
|  4|    0|                    fax|           202-225-3307|
|  4|    1|                  phone|           202-225-5731|
|  5|    0|                    fax|           202-225-3307|
|  5|    1|                  phone|           202-225-5731|
|  6|    0|                    fax|           202-225-3307|
|  6|    1|                  phone|           202-225-5731|
|  7|    0|                    fax|           202-225-3307|
|  7|    1|                  phone|           202-225-5731|
|  8|    0|                    fax|           202-225-3307|
|  8|    1|                  phone|           202-225-5731|
|  9|    0|                    fax|           202-225-3307|
|  9|    1|                  phone|           202-225-5731|
| 10|    0|                    fax|           202-225-6328|
| 10|    1|                  phone|           202-225-4576|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows

Inspect the DynamicFrame that contains rows where ID > 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 11|    0|                    fax|           202-225-6328|
| 11|    1|                  phone|           202-225-4576|
| 11|    2|                twitter|          RepTrentFranks|
| 12|    0|                    fax|           202-225-6328|
| 12|    1|                  phone|           202-225-4576|
| 12|    2|                twitter|          RepTrentFranks|
| 13|    0|                    fax|           202-225-6328|
| 13|    1|                  phone|           202-225-4576|
```

```
| 13|    2|                 twitter|           RepTrentFranks|
| 14|    0|                     fax|             202-225-6328|
| 14|    1|                   phone|             202-225-4576|
| 14|    2|                 twitter|           RepTrentFranks|
| 15|    0|                     fax|             202-225-6328|
| 15|    1|                   phone|             202-225-4576|
| 15|    2|                 twitter|           RepTrentFranks|
| 16|    0|                     fax|             202-225-6328|
| 16|    1|                   phone|             202-225-4576|
| 16|    2|                 twitter|           RepTrentFranks|
| 17|    0|                     fax|             202-225-6328|
| 17|    1|                   phone|             202-225-4576|
+---+-----+----------------------+-------------------------+
only showing top 20 rows

Inspect the merged DynamicFrame that contains the combined rows
+---+-----+----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+-------------------------+
|  1|    0|                     fax|             202-225-3307|
|  1|    1|                   phone|             202-225-5731|
| 20|    0|                     fax|             202-225-5604|
| 20|    1|                   phone|             202-225-6536|
| 20|    2|                 twitter|                USRepLong|
+---+-----+----------------------+-------------------------+
```

relationalize

`relationalize(root_table_name, staging_path, options, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Konvertiert eine `DynamicFrame` in ein Formular, das in eine relationale Datenbank passt. Die Relationalisierung eines `DynamicFrame` ist besonders nützlich, wenn Sie Daten aus einer NoSQL-Umgebung wie DynamoDB in eine relationale Datenbank wie MySQL verschieben möchten.

Die Transformation generiert eine Liste von Frames, indem verschachtelte Spalten aufgehoben und Array-Spalten pivotiert werden. Sie können die pivotierten Array-Spalten mithilfe des Join-Schlüssels, der während der Auflösung der Verschachtelung generiert wird, mit der Stammtabelle verbinden.

- `root_table_name` – Der Name für die Stammtabelle.

- `staging_path` – Der Pfad, unter dem die Methode Partitionen pivotierter Tabellen im CSV-Format speichern kann (optional). Pivotierte Tabellen werden von diesem Pfad gelesen.
- `options` – Ein Wörterbuch der optionalen Parameter.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie Relationalize, um ein verschachteltes Schema in einem **`DynamicFrame`** zu vereinfachen

In diesem Codebeispiel wird die `relationalize`-Methode verwendet, um ein verschachteltes Schema in ein Formular zu vereinfachen, das in eine relationale Datenbank passt.

Beispieldatensatz

Das Beispiel verwendet einen `DynamicFrame` namens `legislators_combined` mit dem folgenden Schema. `legislators_combined` hat mehrere verschachtelte Felder wie `links`, `images` und `contact_details`, die durch die `relationalize`-Transformation vereinfacht werden.

```
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
```

```
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

Beispiel-Code

```
# Example: Use relationalize to flatten
# a nested schema into a format that fits
# into a relational database.
# Replace DOC-EXAMPLE-S3-BUCKET/tmpDir with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
```

```
glueContext = GlueContext(sc)

# Apply relationalize and inspect new tables
legislators_relationalized = legislators_combined.relationalize(
    "l_root", "s3://DOC-EXAMPLE-BUCKET/tmpDir"
)
legislators_relationalized.keys()

# Compare the schema of the contact_details
# nested field to the new relationalized table that
# represents it
legislators_combined.select_fields("contact_details").printSchema()
legislators_relationalized.select("l_root_contact_details").toDF().where(
    "id = 10 or id = 75"
).orderBy(["id", "index"]).show()
```

Output

Mit der folgenden Ausgabe können Sie das Schema des aufgerufenen verschachtelten Felds `contact_details` mit der Tabelle vergleichen, welche die `relationalize`-Transformation erstellt hat. Beachten Sie, dass die Tabellendatensätze mithilfe eines aufgerufenen Fremdschlüssels `id` und einer `index`-Spalte, die die Positionen des Arrays darstellt, eine Verbindung zur Haupttabelle herstellen.

```
dict_keys(['l_root', 'l_root_images', 'l_root_links', 'l_root_other_names',
 'l_root_contact_details', 'l_root_identifiers'])

root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 10|    0|                    fax|             202-225-4160|
| 10|    1|                  phone|             202-225-3436|
| 75|    0|                    fax|             202-225-6791|
| 75|    1|                  phone|             202-225-2861|
| 75|    2|                twitter|               RepSamFarr|
+---+-----+-----------------------+-----------------------+
```

rename_field

`rename_field(oldName, newName, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Benennt ein Feld in diesem `DynamicFrame` um und gibt einen neuen `DynamicFrame` mit dem umbenannten Feld zurück.

- `oldName` – Der vollständiger Pfad zu dem Knoten, den Sie umbenennen möchten.

  Wenn der alte Name Punkte enthält, funktioniert `RenameField` nicht, es sei denn, Sie setzen Backticks darum (`). Um beispielsweise `this.old.name` durch `thisNewName` zu ersetzen, rufen Sie rename_field wie folgt auf:

  ```
  newDyF = oldDyF.rename_field("`this.old.name`", "thisNewName")
  ```

- `newName` – Der neue Name, als vollständiger Pfad.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie rename_field, um Felder in einem **`DynamicFrame`** umzubenennen

In diesem Codebeispiel wird die `rename_field`-Methode verwendet, um Felder in einem `DynamicFrame` umzubenennen. Beachten Sie, dass in dem Beispiel die Methodenverkettung verwendet wird, um mehrere Felder gleichzeitig umzubenennen.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

Beispiel-Code

```
# Example: Use rename_field to rename fields
# in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Inspect the original orgs schema
orgs = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="organizations_json"
)
print("Original orgs schema: ")
orgs.printSchema()

# Rename fields and view the new schema
orgs = orgs.rename_field("id", "org_id").rename_field("name", "org_name")
print("New orgs schema with renamed fields: ")
orgs.printSchema()
```

Output

```
Original orgs schema:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
```

```
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

New orgs schema with renamed fields:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- classification: string
|-- org_id: string
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string
```

resolveChoice

`resolveChoice(specs = None, choice = "" , database = None , table_name = None , transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, catalog_id = None)`

Löst einen Auswahltyp innerhalb dieses `DynamicFrame` auf und gibt den neuen `DynamicFrame` zurück.

- `specs` – Eine Liste der aufzulösenden spezifischen Mehrdeutigkeiten, jede in Form eines Tupels: `(field_path, action)`.

  Es gibt zwei Möglichkeiten für die Verwendung von `resolveChoice`. Zuerst muss das `specs`-Argument verwendet werden, um eine Sequenz bestimmter Felder und deren Auflösung anzugeben. Der andere Modus für `resolveChoice` ist die Nutzung des `choice`-Arguments für die Angabe einer einzelnen Auflösung für alle `ChoiceTypes`.

  Werte für `specs` werden als Tupels angegeben, bestehend aus `(field_path, action)`-Paaren. Der `field_path`-Wert identifiziert ein spezielles mehrdeutiges Element und der `action`-Wert identifiziert die entsprechende Auflösung. Im Folgenden sind die möglichen Aktionen aufgeführt:

  - `cast:`*`type`* – Versucht, alle Werte in den angegebenen Typ umzuwandeln. Beispiel: `cast:int`.
  - `make_cols` – Konvertiert die einzelnen verschiedenen Typen in eine Spalte namens *`columnName_type`*. Er löst eine potenzielle Mehrdeutigkeit durch Abflachen der Daten auf. Wenn `columnA` beispielsweise `int` oder `string` sein könnte, bestünde die Auflösung darin, zwei Spalten mit den Namen `columnA_int` und `columnA_string` im resultierenden `DynamicFrame` zu erzeugen.
  - `make_struct` – Löst eine potenzielle Mehrdeutigkeit durch Verwendung einer `struct`, um die Daten darzustellen. Wenn beispielsweise Daten in einer Spalte `int` oder `string` sein könnten, wird durch Verwendung der `make_struct`-Aktion eine Spalte von Strukturen im resultierenden `DynamicFrame` erzeugt. Jede Struktur enthält sowohl einen `int` als auch einen `string`.
  - `project:`*`type`* – Löst eine potenzielle Mehrdeutigkeit durch Projizierung aller Daten auf einen der möglichen Datentypen. Wenn etwa Daten in einer Spalte `int` oder `string` sein könnten, wird mithilfe einer `project:string`-Aktion eine Spalte im resultierenden `DynamicFrame` erzeugt, in der alle `int`-Werte in Zeichenfolgen konvertiert wurden.

Wenn `field_path` ein Array identifiziert, platzieren Sie leere eckige Klammern hinter dem Namen des Arrays, um eine Mehrdeutigkeit zu vermeiden. Angenommen, Sie arbeiten mit Daten, die wie folgt strukturiert sind:

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
]
```

Sie können anstelle der Zeichenfolgenversion die numerische Version des Preises auswählen, indem Sie den `field_path` auf `"myList[].price"` und die `action` auf `"cast:double"` setzen.

> **Note**
>
> Sie können nur einen der Parameter `specs` und `choice` verwenden. Wenn der `specs`-Parameter nicht `None` ist, dann muss der `choice`-Parameter eine leere Zeichenfolge sein. Wenn umgekehrt der `choice`-Parameter keine leere Zeichenfolge ist, dann muss der `specs`-Parameter `None` sein.

- `choice` – Gibt eine einzelne Auflösung für alle `ChoiceTypes` an. Sie können dies verwenden, wenn die vollständige Liste der `ChoiceTypes` vor der Laufzeit unbekannt ist. Zusätzlich zu den soeben für `specs` aufgeführten Aktionen unterstützt dieses Argument noch die folgende Aktion:
  - `match_catalog` – Versucht jeden `ChoiceType` in einen entsprechenden Typ in der angegebenen Data-Catalog-Tabelle umzuwandeln.
- `database` – Die Data-Catalog-Datenbank für die Verwendung mit der `match_catalog`-Aktion.
- `table_name` – Die Data-Catalog-Tabelle für die Verwendung mit der `match_catalog`-Aktion.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Null als Standard gibt an, dass der Prozess keinen Fehler ausgeben sollte.
- `catalog_id` – Die Katalog-ID des Data Catalogs, auf den zugegriffen wird (die Konto-ID des Data Catalogs). Wenn auf `None` (Standardwert) festgelegt, wird die Katalog-ID des aufrufenden Kontos verwendet.

Beispiel: Verwenden Sie ResolveChoice, um eine Spalte zu behandeln, die mehrere Typen enthält

In diesem Codebeispiel wird die `resolveChoice`-Methode verwendet, um anzugeben, wie mit einer `DynamicFrame`-Spalte umgegangen werden soll, die Werte mehrerer Typen enthält. Das Beispiel zeigt zwei gängige Methoden zur Behandlung einer Spalte mit unterschiedlichen Typen:

- Umwandeln der Spalte in einen einzelnen Datentyp.
- Beibehalten aller Typen in separaten Spalten.

Beispieldatensatz

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Datenvorbereitung mit ResolveChoice, Lambda und ApplyMapping und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

Das Beispiel verwendet einen `DynamicFrame` mit dem Namen `medicare` mit dem folgenden Schema:

```
root
|-- drg definition: string
|-- provider id: choice
|    |-- long
|    |-- string
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
```

```
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string
```

Beispiel-Code

```
# Example: Use resolveChoice to handle
# a column that contains multiple types

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input data and inspect the "provider id" column
medicare = glueContext.create_dynamic_frame.from_catalog(
    database="payments", table_name="medicare_hospital_provider_csv"
)
print("Inspect the provider id column:")
medicare.toDF().select("provider id").show()

# Cast provider id to type long
medicare_resolved_long = medicare.resolveChoice(specs=[("provider id", "cast:long")])
print("Schema after casting provider id to type long:")
medicare_resolved_long.printSchema()
medicare_resolved_long.toDF().select("provider id").show()

# Create separate columns
# for each provider id type
medicare_resolved_cols = medicare.resolveChoice(choice="make_cols")
print("Schema after creating separate columns for each type:")
medicare_resolved_cols.printSchema()
medicare_resolved_cols.toDF().select("provider id_long", "provider id_string").show()
```

Output

```
Inspect the 'provider id' column:
+-----------+
|provider id|
+-----------+
```

```
|   [10001,]|
|   [10005,]|
|   [10006,]|
|   [10011,]|
|   [10016,]|
|   [10023,]|
|   [10029,]|
|   [10033,]|
|   [10039,]|
|   [10040,]|
|   [10046,]|
|   [10055,]|
|   [10056,]|
|   [10078,]|
|   [10083,]|
|   [10085,]|
|   [10090,]|
|   [10092,]|
|   [10100,]|
|   [10103,]|
+-----------+
only showing top 20 rows

Schema after casting 'provider id' to type long:
root
|-- drg definition: string
|-- provider id: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string

+-----------+
|provider id|
+-----------+
|      10001|
|      10005|
|      10006|
```

```
|      10011|
|      10016|
|      10023|
|      10029|
|      10033|
|      10039|
|      10040|
|      10046|
|      10055|
|      10056|
|      10078|
|      10083|
|      10085|
|      10090|
|      10092|
|      10100|
|      10103|
+-----------+
only showing top 20 rows

Schema after creating separate columns for each type:
root
|-- drg definition: string
|-- provider id_string: string
|-- provider id_long: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string

+----------------+------------------+
|provider id_long|provider id_string|
+----------------+------------------+
|           10001|              null|
|           10005|              null|
|           10006|              null|
|           10011|              null|
|           10016|              null|
```

```
|           10023|              null|
|           10029|              null|
|           10033|              null|
|           10039|              null|
|           10040|              null|
|           10046|              null|
|           10055|              null|
|           10056|              null|
|           10078|              null|
|           10083|              null|
|           10085|              null|
|           10090|              null|
|           10092|              null|
|           10100|              null|
|           10103|              null|
+----------------+------------------+
only showing top 20 rows
```

select_fields

`select_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Gibt einen neuen `DynamicFrame` zurück, der die ausgewählten Felder enthält.

- `paths` – Eine Liste von Zeichenfolgen. Jede Zeichenfolgen ist ein vollständiger Pfad zu einem Knoten der obersten Ebene ist, den Sie auswählen möchten.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden von select_fields zum Erstellen eines neuen **`DynamicFrame`** mit ausgewählten Feldern

Im folgenden Beispielcode wird gezeigt, wie Sie die `select_fields`-Methode zum Erstellen eines neuen `DynamicFrame` mit einer ausgewählten Liste von Feldern aus einem vorhandenen `DynamicFrame` verwenden können.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use select_fields to select specific fields from a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Create a new DynamicFrame with chosen fields
names = persons.select_fields(paths=["family_name", "given_name"])
print("Schema for the names DynamicFrame, created with select_fields:")
names.printSchema()
names.toDF().show()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
```

```
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the names DynamicFrame:
root
|-- family_name: string
|-- given_name: string

+-----------+----------+
|family_name|given_name|
+-----------+----------+
|    Collins|   Michael|
|   Huizenga|      Bill|
|    Clawson|    Curtis|
|    Solomon|    Gerald|
|     Rigell|    Edward|
|      Crapo|   Michael|
|      Hutto|      Earl|
```

```
|      Ertel|     Allen|
|     Minish|    Joseph|
|    Andrews|    Robert|
|     Walden|      Greg|
|      Kazen|   Abraham|
|     Turner|   Michael|
|      Kolbe|     James|
|  Lowenthal|      Alan|
|    Capuano|   Michael|
|   Schrader|      Kurt|
|     Nadler|   Jerrold|
|     Graves|       Tom|
|   McMillan|      John|
+-----------+----------+
only showing top 20 rows
```

simplify_ddb_json

`simplify_ddb_json(): DynamicFrame`

Vereinfacht verschachtelte Spalten in a`DynamicFrame`, die sich speziell in der DynamoDB-JSON-Struktur befinden, und gibt eine neue vereinfachte Spalte zurück. `DynamicFrame` Wenn ein Listentyp mehrere Typen oder einen Map-Typ enthält, werden die Elemente in der Liste nicht vereinfacht. Beachten Sie, dass dies ein bestimmter Transformationstyp ist, der sich anders verhält als die reguläre `unnest` Transformation und erfordert, dass sich die Daten bereits in der DynamoDB-JSON-Struktur befinden. Weitere Informationen finden Sie unter DYNAMODB JSON.

Das Schema eines Vorgangs zum Lesen eines Exports mit der DynamoDB-JSON-Struktur könnte beispielsweise wie folgt aussehen:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
```

```
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

Die `simplify_ddb_json()`-Transformation würde dies folgendermaßen umwandeln:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

Beispiel: Verwenden Sie simplify_ddb_json, um eine DynamoDB-JSON-Vereinfachung aufzurufen

In diesem Codebeispiel wird die `simplify_ddb_json` Methode verwendet, um den AWS Glue DynamoDB-Exportconnector zu verwenden, eine DynamoDB-JSON-Vereinfachung aufzurufen und die Anzahl der Partitionen auszudrucken.

Beispiel-Code

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
sc = SparkContext()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "dynamodb",
    connection_options = {
        'dynamodb.export': 'ddb',
        'dynamodb.tableArn': '<table arn>',
        'dynamodb.s3.bucket': '<bucket name>',
        'dynamodb.s3.prefix': '<bucket prefix>',
        'dynamodb.s3.bucketOwner': '<account_id of bucket>'
    }
)
simplified = dynamicFrame.simplify_ddb_json()
print(simplified.getNumPartitions())
```

spigot

`spigot(path, options={})`

Schreibt Beispieldatensätze an ein bestimmtes Ziel, damit Sie die von Ihrem Auftrag durchgeführten Transformationen überprüfen können.

- `path` – Der Pfad zum Ziel, an das geschrieben werden soll (erforderlich).
- `options` – Schlüssel-Wert-Paare, die Optionen angeben (optional). Die `"topk"`-Option gibt an, dass die ersten k-Datensätze geschrieben werden sollen. Die `"prob"`-Option gibt an, wie wahrscheinlich es ist (in Form einer Dezimalzahl), dass ein bestimmter Datensatz ausgewählt wird. Sie können sie verwenden, um Datensätze auszuwählen, die geschrieben werden sollen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

Beispiel: Verwenden Sie Spigot, um Beispielfelder von einem **`DynamicFrame`** an Amazon S3 zu schreiben

In diesem Codebeispiel wird die `spigot`-Methode verwendet, um nach der Anwendung der `select_fields`-Transformation Beispieldatensätze in einen Amazon-S3-Bucket zu schreiben.

Beispieldatensatz

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

Das Beispiel verwendet einen `DynamicFrame` mit dem Namen `persons` mit dem folgenden Schema:

```
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Beispiel-Code

```
# Example: Use spigot to write sample records
# to a destination during a transformation
# from pyspark.context import SparkContext.
# Replace DOC-EXAMPLE-BUCKET with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load table data into a DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)

# Perform the select_fields on the DynamicFrame
persons = persons.select_fields(paths=["family_name", "given_name", "birth_date"])

# Use spigot to write a sample of the transformed data
# (the first 10 records)
spigot_output = persons.spigot(
    path="s3://DOC-EXAMPLE-BUCKET", options={"topk": 10}
)
# Example: Use spigot to write sample records
# to a destination during a transformation
# from pyspark.context import SparkContext.
# Replace DOC-EXAMPLE-BUCKET with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load table data into a DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)

# Perform the select_fields on the DynamicFrame
```

```
persons = persons.select_fields(paths=["family_name", "given_name", "birth_date"])

# Use spigot to write a sample of the transformed data
# (the first 10 records)
spigot_output = persons.spigot(
    path="s3://DOC-EXAMPLE-BUCKET", options={"topk": 10}
)
```

Output

Im Folgenden finden Sie ein Beispiel für die Daten, die `spigot` an Amazon S3 schreibt. Da der Beispielcode `options={"topk": 10}` angegeben hat, enthalten die Beispieldaten die ersten zehn Datensätze.

```
{"family_name":"Collins","given_name":"Michael","birth_date":"1944-10-15"}
{"family_name":"Huizenga","given_name":"Bill","birth_date":"1969-01-31"}
{"family_name":"Clawson","given_name":"Curtis","birth_date":"1959-09-28"}
{"family_name":"Solomon","given_name":"Gerald","birth_date":"1930-08-14"}
{"family_name":"Rigell","given_name":"Edward","birth_date":"1960-05-28"}
{"family_name":"Crapo","given_name":"Michael","birth_date":"1951-05-20"}
{"family_name":"Hutto","given_name":"Earl","birth_date":"1926-05-12"}
{"family_name":"Ertel","given_name":"Allen","birth_date":"1937-11-07"}
{"family_name":"Minish","given_name":"Joseph","birth_date":"1916-09-01"}
{"family_name":"Andrews","given_name":"Robert","birth_date":"1957-08-04"}
```

split_fields

`split_fields(paths, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Gibt einen neuen `DynamicFrameCollection` zurück, der zwei `DynamicFrames` enthält. Der erste `DynamicFrame` enthält alle Knoten, die abgespaltet wurden, und der zweite enthält die Knoten, die verbleiben.

- `paths` – Eine Liste von Zeichenfolgen, von denen jede ein vollständiger Pfad zu einem Knoten ist, den Sie in einen neuen `DynamicFrame` teilen möchten.
- `name1` – Eine Namenszeichenfolge für den `DynamicFrame`, der abgespaltet ist.
- `name2` – Eine Namenszeichenfolge für den `DynamicFrame`, der verbleibt, nachdem die angegebenen Knoten abgespaltet wurden.

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie split_fields, um ausgewählte Felder in einen separaten **`DynamicFrame`** aufzuteilen

In diesem Codebeispiel wird die `split_fields`-Methode verwendet, um eine Liste von angegebenen Feldern in einen separaten `DynamicFrame` aufzuteilen.

Beispieldatensatz

Das Beispiel verwendet einen `DynamicFrame` namens `l_root_contact_details`, der aus einer Sammlung mit dem Namen `legislators_relationalized` stammt.

`l_root_contact_details` hat das folgende Schema und folgende Einträge.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
...
```

Beispiel-Code

```
# Example: Use split_fields to split selected
# fields into a separate DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input DynamicFrame and inspect its schema
frame_to_split = legislators_relationalized.select("l_root_contact_details")
print("Inspect the input DynamicFrame schema:")
frame_to_split.printSchema()

# Split id and index fields into a separate DynamicFrame
split_fields_collection = frame_to_split.split_fields(["id", "index"], "left", "right")

# Inspect the resulting DynamicFrames
print("Inspect the schemas of the DynamicFrames created with split_fields:")
split_fields_collection.select("left").printSchema()
split_fields_collection.select("right").printSchema()
```

Output

```
Inspect the input DynamicFrame's schema:
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

Inspect the schemas of the DynamicFrames created with split_fields:
root
|-- id: long
|-- index: int

root
|-- contact_details.val.type: string
|-- contact_details.val.value: string
```

split_rows

`split_rows(comparison_dict, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Teilt eine oder mehrere Zeilen in einem `DynamicFrame` auf einen neuen `DynamicFrame` auf.

Die Methode gibt einen neuen `DynamicFrameCollection` zurück, der zwei `DynamicFrames` enthält. Der erste `DynamicFrame` enthält alle Zeilen, die abgespaltet wurden, und der zweite enthält die Zeilen, die verbleiben.

- `comparison_dict` – Ein Wörterbuch, in dem der Schlüssel ein Pfad zu einer Spalte ist und der Wert ein weiteres Wörterbuch für das Mapping von Vergleichsoperatoren zu Werten ist, mit denen der Spaltenwert verglichen wird. Beispielsweise spaltet `{"age": {">": 10, "<": 20}}` alle Zeilen ab, deren Wert in der Spalte Alter größer als 10 und kleiner als 20 ist.
- `name1` – Eine Namenszeichenfolge für den `DynamicFrame`, der abgespaltet ist.
- `name2` – Eine Namenszeichenfolge für den `DynamicFrame`, der verbleibt, nachdem die angegebenen Knoten abgespaltet wurden.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie split_rows, um Zeilen in einem **`DynamicFrame`** abzuspalten

In diesem Codebeispiel wird die `split_rows`-Methode verwendet, um Zeilen in einem `DynamicFrame` auf der Grundlage des `id`-Feldwerts abzuspalten.

Beispieldatensatz

Das Beispiel verwendet einen `DynamicFrame` namens `l_root_contact_details`, der aus einer Sammlung mit dem Namen `legislators_relationalized` ausgewählt wird.

`l_root_contact_details` hat das folgende Schema und folgende Einträge.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
|  3|    2|                twitter|          MikeRossUpdates|
|  4|    0|                    fax|             202-225-1314|
|  4|    1|                  phone|             202-225-3772|
|  4|    2|                twitter|          MikeRossUpdates|
|  5|    0|                    fax|             202-225-1314|
|  5|    1|                  phone|             202-225-3772|
|  5|    2|                twitter|          MikeRossUpdates|
|  6|    0|                    fax|             202-225-1314|
|  6|    1|                  phone|             202-225-3772|
|  6|    2|                twitter|          MikeRossUpdates|
|  7|    0|                    fax|             202-225-1314|
|  7|    1|                  phone|             202-225-3772|
|  7|    2|                twitter|          MikeRossUpdates|
|  8|    0|                    fax|             202-225-1314|
+---+-----+-----------------------+-----------------------+
```

Beispiel-Code

```
# Example: Use split_rows to split up
# rows in a DynamicFrame based on value

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
```

```
glueContext = GlueContext(sc)

# Retrieve the DynamicFrame to split
frame_to_split = legislators_relationalized.select("l_root_contact_details")

# Split up rows by ID
split_rows_collection = frame_to_split.split_rows({"id": {">": 10}}, "high", "low")

# Inspect the resulting DynamicFrames
print("Inspect the DynamicFrame that contains IDs < 10")
split_rows_collection.select("low").toDF().show()
print("Inspect the DynamicFrame that contains IDs > 10")
split_rows_collection.select("high").toDF().show()
```

Output

```
Inspect the DynamicFrame that contains IDs < 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
|  3|    2|                twitter|          MikeRossUpdates|
|  4|    0|                    fax|             202-225-1314|
|  4|    1|                  phone|             202-225-3772|
|  4|    2|                twitter|          MikeRossUpdates|
|  5|    0|                    fax|             202-225-1314|
|  5|    1|                  phone|             202-225-3772|
|  5|    2|                twitter|          MikeRossUpdates|
|  6|    0|                    fax|             202-225-1314|
|  6|    1|                  phone|             202-225-3772|
|  6|    2|                twitter|          MikeRossUpdates|
|  7|    0|                    fax|             202-225-1314|
|  7|    1|                  phone|             202-225-3772|
|  7|    2|                twitter|          MikeRossUpdates|
|  8|    0|                    fax|             202-225-1314|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows
```

```
Inspect the DynamicFrame that contains IDs > 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 11|    0|                  phone|             202-225-5476|
| 11|    1|                twitter|            RepDavidYoung|
| 12|    0|                  phone|             202-225-4035|
| 12|    1|                twitter|           RepStephMurphy|
| 13|    0|                    fax|             202-226-0774|
| 13|    1|                  phone|             202-225-6335|
| 14|    0|                    fax|             202-226-0774|
| 14|    1|                  phone|             202-225-6335|
| 15|    0|                    fax|             202-226-0774|
| 15|    1|                  phone|             202-225-6335|
| 16|    0|                    fax|             202-226-0774|
| 16|    1|                  phone|             202-225-6335|
| 17|    0|                    fax|             202-226-0774|
| 17|    1|                  phone|             202-225-6335|
| 18|    0|                    fax|             202-226-0774|
| 18|    1|                  phone|             202-225-6335|
| 19|    0|                    fax|             202-226-0774|
| 19|    1|                  phone|             202-225-6335|
| 20|    0|                    fax|             202-226-0774|
| 20|    1|                  phone|             202-225-6335|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows
```

unbox

unbox(path, format, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, **options)

Entpackt (formatiert neu) ein Zeichenfolgefeld in einem `DynamicFrame` und gibt einen neuen `DynamicFrame` zurück, der die entpackten `DynamicRecords` enthält.

Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Apache Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.

- `path` – Ein vollständiger Pfad zu dem Zeichenfolgeknoten, den Sie entpacken möchten.

- `format` – Eine Formatspezifikation (optional). Sie verwenden dies für einen Amazon S3 oder AWS Glue Verbindung, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `options` – Eine oder mehrere der folgenden Aktionen:
  - `separator` – Eine Zeichenfolge mit Trennzeichen.
  - `escaper` – Eine Zeichenfolge mit Escape-Zeichen.
  - `skipFirst` – Ein Boolescher Wert, der angibt, ob die erste Instance übersprungen werden soll.
  - `withSchema` – Eine Zeichenfolge, die eine JSON-Darstellung des Schemas des Knotens enthält. Das Format der JSON-Darstellung eines Schemas wird durch die Ausgabe von `StructType.json()` definiert.
  - `withHeader` – Ein Boolescher Wert, der angibt, ob ein Header enthalten ist.

Beispiel: Verwenden Sie unbox, um ein Zeichenkettenfeld in eine Struktur zu entpacken

In diesem Codebeispiel wird die `unbox`-Methode verwendet, um ein Zeichenfolgenfeld in einem `DynamicFrame` in ein Feld vom Typ Struktur zu entpacken oder neu zu formatieren.

Beispieldatensatz

Das Beispiel verwendet einen `DynamicFrame` mit dem Namen `mapped_with_string` mit dem folgenden Schema und den folgenden Einträgen.

Beachten Sie das Feld mit dem Namen `AddressString`. Dies ist das Feld, das das Beispiel in eine Struktur entpackt.

```
root
```

```
|-- Average Total Payments: string
|-- AddressString: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|              $5777.24|{"Street": "1108 ...|              $32963.07|039 -
 EXTRACRANIA...|                $4763.73|                          AL - Dothan|[36301,
 DOTHAN, [...|      10001|              91|SOUTHEAST ALABAMA...|
|              $5787.57|{"Street": "2505 ...|              $15131.85|039 -
 EXTRACRANIA...|                $4976.71|                      AL - Birmingham|[35957,
 BOAZ, [25...|      10005|              14|MARSHALL MEDICAL ...|
|              $5434.95|{"Street": "205 M...|              $37560.37|039 -
 EXTRACRANIA...|                $4453.79|                      AL - Birmingham|[35631,
 FLORENCE,...|      10006|              24|ELIZA COFFEE MEMO...|
|              $5417.56|{"Street": "50 ME...|              $13998.28|039 -
 EXTRACRANIA...|                $4129.16|                      AL - Birmingham|[35235,
 BIRMINGHA...|      10011|              25|   ST VINCENT'S EAST|
...
```

Beispiel-Code

```
# Example: Use unbox to unbox a string field
# into a struct in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

unboxed = mapped_with_string.unbox("AddressString", "json")
unboxed.printSchema()
unboxed.toDF().show()
```

Output

```
root
|-- Average Total Payments: string
|-- AddressString: struct
|    |-- Street: string
|    |-- City: string
|    |-- State: string
|    |-- Zip.Code: string
|    |-- Array: array
|    |    |-- element: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+-------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
```

```
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
|              $5777.24|[1108 ROSS CLARK ...|              $32963.07|039 -
 EXTRACRANIA...|                 $4763.73|                         AL - Dothan|[36301,
 DOTHAN, [...|      10001|              91|SOUTHEAST ALABAMA...|
|              $5787.57|[2505 U S HIGHWAY...|              $15131.85|039 -
 EXTRACRANIA...|                 $4976.71|                     AL - Birmingham|[35957,
 BOAZ, [25...|      10005|              14|MARSHALL MEDICAL ...|
|              $5434.95|[205 MARENGO STRE...|              $37560.37|039 -
 EXTRACRANIA...|                 $4453.79|                     AL - Birmingham|[35631,
 FLORENCE,...|      10006|              24|ELIZA COFFEE MEMO...|
|              $5417.56|[50 MEDICAL PARK ...|              $13998.28|039 -
 EXTRACRANIA...|                 $4129.16|                     AL - Birmingham|[35235,
 BIRMINGHA...|      10011|              25|   ST VINCENT'S EAST|
|              $5658.33|[1000 FIRST STREE...|              $31633.27|039 -
 EXTRACRANIA...|                 $4851.44|                     AL - Birmingham|[35007,
 ALABASTER...|      10016|              18|SHELBY BAPTIST ME...|
|              $6653.80|[2105 EAST SOUTH ...|              $16920.79|039 -
 EXTRACRANIA...|                 $5374.14|                     AL - Montgomery|[36116,
 MONTGOMER...|      10023|              67|BAPTIST MEDICAL C...|
|              $5834.74|[2000 PEPPERELL P...|              $11977.13|039 -
 EXTRACRANIA...|                 $4761.41|                     AL - Birmingham|[36801,
 OPELIKA, ...|      10029|              51|EAST ALABAMA MEDI...|
|              $8031.12|[619 SOUTH 19TH S...|              $35841.09|039 -
 EXTRACRANIA...|                 $5858.50|                     AL - Birmingham|[35233,
 BIRMINGHA...|      10033|              32|UNIVERSITY OF ALA...|
|              $6113.38|[101 SIVLEY RD, H...|              $28523.39|039 -
 EXTRACRANIA...|                 $5228.40|                     AL - Huntsville|[35801,
 HUNTSVILL...|      10039|             135| HUNTSVILLE HOSPITAL|
|              $5541.05|[1007 GOODYEAR AV...|              $75233.38|039 -
 EXTRACRANIA...|                 $4386.94|                     AL - Birmingham|[35903,
 GADSDEN, ...|      10040|              34|GADSDEN REGIONAL ...|
|              $5461.57|[600 SOUTH THIRD ...|              $67327.92|039 -
 EXTRACRANIA...|                 $4493.57|                     AL - Birmingham|[35901,
 GADSDEN, ...|      10046|              14|RIVERVIEW REGIONA...|
|              $5356.28|[4370 WEST MAIN S...|              $39607.28|039 -
 EXTRACRANIA...|                 $4408.20|                         AL - Dothan|[36305,
 DOTHAN, [...|      10055|              45|    FLOWERS HOSPITAL|
```

```
|              $5374.65|[810 ST VINCENT'S...|              $22862.23|039 -
 EXTRACRANIA...|                $4186.02|                    AL - Birmingham|[35205,
 BIRMINGHA...|      10056|              43|ST VINCENT'S BIRM...|
|              $5366.23|[400 EAST 10TH ST...|              $31110.85|039 -
 EXTRACRANIA...|                $4376.23|                    AL - Birmingham|[36207,
 ANNISTON,...|      10078|              21|NORTHEAST ALABAMA...|
|              $5282.93|[1613 NORTH MCKEN...|              $25411.33|039 -
 EXTRACRANIA...|                $4383.73|                       AL - Mobile|[36535,
 FOLEY, [1...|      10083|              15|SOUTH BALDWIN REG...|
|              $5676.55|[1201 7TH STREET ...|               $9234.51|039 -
 EXTRACRANIA...|                $4509.11|                    AL - Huntsville|[35609,
 DECATUR, ...|      10085|              27|DECATUR GENERAL H...|
|              $5930.11|[6801 AIRPORT BOU...|              $15895.85|039 -
 EXTRACRANIA...|                $3972.85|                       AL - Mobile|[36608,
 MOBILE, [...|      10090|              27| PROVIDENCE HOSPITAL|
|              $6192.54|[809 UNIVERSITY B...|              $19721.16|039 -
 EXTRACRANIA...|                $5179.38|                    AL - Tuscaloosa|[35401,
 TUSCALOOS...|      10092|              31|D C H REGIONAL ME...|
|              $4968.00|[750 MORPHY AVENU...|              $10710.88|039 -
 EXTRACRANIA...|                $3898.88|                       AL - Mobile|[36532,
 FAIRHOPE,...|      10100|              18|     THOMAS HOSPITAL|
|              $5996.00|[701 PRINCETON AV...|              $51343.75|039 -
 EXTRACRANIA...|                $4962.45|                    AL - Birmingham|[35211,
 BIRMINGHA...|      10103|              33|BAPTIST MEDICAL C...|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
only showing top 20 rows
```

union

union(frame1, frame2, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Union zwei DynamicFrames. Gibt zurück DynamicFrame , die alle Datensätze aus beiden Eingaben enthält DynamicFrames. Diese Transformation kann aus der Vereinigung von zwei DataFrames mit äquivalenten Daten unterschiedliche Ergebnisse liefern. Wenn Sie das DataFrame Union-Verhalten von Spark benötigen, sollten Sie es verwenden`toDF`.

- `frame1`— Zuerst DynamicFrame zur Vereinigung.
- `frame2`— An zweiter Stelle DynamicFrame nach der Gewerkschaft.

- `transformation_ctx` – (optional) Eine eindeutige Zeichenfolge zur Identifikation von Status-/Zustandsinformationen
- `info` – (optional) Eine Zeichenfolge im Zusammenhang mit Fehlern bei der Transformation
- `stageThreshold` – (optional) Maximale Anzahl von Fehlern bei der Transformation, bis die Verarbeitung aufgrund eines Fehlers fehlschlägt
- `totalThreshold` – (optional) Maximale Anzahl von Fehlern insgesamt, bis die Verarbeitung aufgrund eines Fehlers fehlschlägt.

unnest

`unnest(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Löst verschachtelte Objekte in einem `DynamicFrame` auf, wodurch sie zu Objekten der obersten Ebene werden, und gibt einen neuen, nicht verschachtelten `DynamicFrame` zurück.

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional). Die Standardeinstellung ist Null, was darauf hinweist, dass der Prozess nicht fehlerhaft sein sollte.

Beispiel: Verwenden Sie unnest, um verschachtelte Felder in Felder der obersten Ebene umzuwandeln

In diesem Codebeispiel wird die `unnest`-Methode verwendet, um alle verschachtelten Felder in einem `DynamicFrame` zu Feldern der obersten Ebene zu vereinfachen.

Beispieldatensatz

Das Beispiel verwendet einen `DynamicFrame` mit dem Namen `mapped_medicare` mit dem folgenden Schema. Beachten Sie, dass das `Address`-Feld das einzige Feld ist, das verschachtelte Daten enthält.

```
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

Beispiel-Code

```
# Example: Use unnest to unnest nested
# objects in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Unnest all nested fields
unnested = mapped_medicare.unnest()
unnested.printSchema()
```

Output

```
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address.Zip.Code: string
```

```
|-- Address.City: string
|-- Address.Array: array
|    |-- element: string
|-- Address.State: string
|-- Address.Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

unnest_ddb_json

Hebt die Verschachtelung der Spalten in einem `DynamicFrame` auf, die sich speziell in der DynamoDB-JSON-Struktur befinden, und gibt einen neuen, nicht verschachtelten `DynamicFrame` zurück. Bei Spalten, die aus einem Array von Strukturtypen bestehen, wird die Verschachtelung nicht aufgehoben. Dies ist ein spezieller Typ der Transformation zum Aufheben der Verschachtelung, der sich anders verhält als die reguläre `unnest`-Transformation und erfordert, dass sich die Daten bereits in der DynamoDB-JSON-Struktur befinden. Weitere Informationen finden Sie unter DYNAMODB JSON.

`unnest_ddb_json(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die den Fehlermeldungen für diese Transformation zugeordnet werden soll (optional).
- `stageThreshold` – Die Anzahl der aufgetretenen Fehler während dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional: Null als Standard gibt an, dass der Prozess keinen Fehler ausgeben sollte).
- `totalThreshold` – Die Anzahl der aufgetretenen Fehler bis einschließlich dieser Transformation bei der der Prozess einen Fehler ausgeben sollte (optional: Null als Standard gibt an, dass der Prozess keinen Fehler ausgeben sollte).

Das Schema eines Vorgangs zum Lesen eines Exports mit der DynamoDB-JSON-Struktur könnte beispielsweise wie folgt aussehen:

```
root
```

```
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
|    |    |-- N: string
|    |-- ColD: struct
|    |    |-- L: array
|    |    |    |-- element: null
```

Die `unnest_ddb_json()`-Transformation würde dies folgendermaßen umwandeln:

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

Das folgende Codebeispiel zeigt, wie Sie den AWS Glue DynamoDB-Exportconnector verwenden, einen DynamoDB-JSON-Unnest aufrufen und die Anzahl der Partitionen ausgeben:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": "<test_source>",
        "dynamodb.s3.bucket": "<bucket name>",
        "dynamodb.s3.prefix": "<bucket prefix>",
        "dynamodb.s3.bucketOwner": "<account_id>",
    }
```

```
)
unnested = dynamicFrame.unnest_ddb_json()
print(unnested.getNumPartitions())

job.commit()
```

write

`write(connection_type, connection_options, format, format_options, accumulator_size)`

Bezieht einen DataSink(Objekt) des angegebenen Verbindungstyps vom GlueContext Klasse dieses `DynamicFrame` und verwendet ihn zum Formatieren und Schreiben des Inhalts dieses `DynamicFrame`. Gibt den neuen `DynamicFrame` wie angegeben formatiert und geschrieben zurück.

- `connection_type` – Der zu verwendende Verbindungstyp. Gültige Werte sind `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` und `oracle`.
- `connection_options` – Die zu verwendende Verbindungsoption (optional). Für den `connection_type` `s3` ist ein Amazon-S3-Pfad definiert.

  ```
  connection_options = {"path": "s3://aws-glue-target/temp"}
  ```

  Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

  > **Warning**
  >
  > Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

  ```
  connection_options = {"url": "jdbc-url/database", "user": "username",
   "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
  path"}
  ```

- `format` – Eine Formatspezifikation (optional). Dies wird für einen Amazon Simple Storage Service (Amazon S3) verwendet oder AWS Glue Verbindung, die mehrere Formate unterstützt.

Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.

- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `accumulator_size` – Die zu verwendende akkumulierbare Größe in Bytes (optional).

– Fehler –

- assertErrorThreshold
- errorsAsDynamicRahmen
- errorsCount
- stageErrorsCount

assertErrorThreshold

`assertErrorThreshold( )` – Eine Assertion für Fehler in den Transformationen, die diesen `DynamicFrame` erstellt hat. Gibt eine `Exception` von dem zugrunde liegenden `DataFrame` zurück.

errorsAsDynamicRahmen

`errorsAsDynamicFrame( )` – Gibt einen `DynamicFrame` zurück, der verschachtelte Fehlerdatensätze enthält.

Beispiel: Verwenden Sie errorsAsDynamic Frame, um Fehlerdatensätze anzuzeigen

Im folgenden Beispielcode wird gezeigt, wie Sie die `errorsAsDynamicFrame`-Methode zum Anzeigen eines Fehlerdatensatzes für einen `DynamicFrame` verwenden.

Beispieldatensatz

Das Beispiel verwendet den folgenden Datensatz, den Sie als JSON auf Amazon S3 hochladen können. Beachten Sie, dass der zweite Datensatz falsch formatiert ist. Fehlerhaft formatierte Daten führen normalerweise zu einer Störung der Dateianalyse, wenn Sie SparkSQL verwenden. Allerdings erkennt `DynamicFrame` Fehlbildungsprobleme und wandelt fehlerhafte Zeilen in Fehlerdatensätze um, die Sie individuell behandeln können.

```
{"id": 1, "name": "george", "surname": "washington", "height": 178}
```

```
{"id": 2, "name": "benjamin", "surname": "franklin",
{"id": 3, "name": "alexander", "surname": "hamilton", "height": 171}
{"id": 4, "name": "john", "surname": "jay", "height": 190}
```

Beispiel-Code

```
# Example: Use errorsAsDynamicFrame to view error records.
# Replace s3://DOC-EXAMPLE-S3-BUCKET/error_data.json with your location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create errors DynamicFrame, view schema
errors = glueContext.create_dynamic_frame.from_options(
    "s3", {"paths": ["s3://DOC-EXAMPLE-S3-BUCKET/error_data.json"]}, "json"
)
print("Schema of errors DynamicFrame:")
errors.printSchema()

# Show that errors only contains valid entries from the dataset
print("errors contains only valid records from the input dataset (2 of 4 records)")
errors.toDF().show()

# View errors
print("Errors count:", str(errors.errorsCount()))
print("Errors:")
errors.errorsAsDynamicFrame().toDF().show()

# View error fields and error data
error_record = errors.errorsAsDynamicFrame().toDF().head()

error_fields = error_record["error"]
print("Error fields: ")
print(error_fields.asDict().keys())

print("\nError record data:")
for key in error_fields.asDict().keys():
    print("\n", key, ": ", str(error_fields[key]))
```

Output

```
Schema of errors DynamicFrame:
root
|-- id: int
|-- name: string
|-- surname: string
|-- height: int

errors contains only valid records from the input dataset (2 of 4 records)
+---+------+----------+------+
| id|  name|   surname|height|
+---+------+----------+------+
|  1|george|washington|   178|
|  4|  john|       jay|   190|
+---+------+----------+------+

Errors count: 1
Errors:
+--------------------+
|               error|
+--------------------+
|[[  File "/tmp/20...|
+--------------------+

Error fields:
dict_keys(['callsite', 'msg', 'stackTrace', 'input', 'bytesread', 'source',
 'dynamicRecord'])

Error record data:

 callsite :  Row(site='  File "/tmp/2060612586885849088", line 549, in <module>\n
 sys.exit(main())\n  File "/tmp/2060612586885849088", line 523, in main\n    response
 = handler(content)\n  File "/tmp/2060612586885849088", line 197, in execute_request
\n    result = node.execute()\n  File "/tmp/2060612586885849088", line 103, in
 execute\n    exec(code, global_dict)\n  File "<stdin>", line 10, in <module>\n
 File "/opt/amazon/lib/python3.6/site-packages/awsglue/dynamicframe.py", line 625, in
 from_options\n    format_options, transformation_ctx, push_down_predicate, **kwargs)\n
  File "/opt/amazon/lib/python3.6/site-packages/awsglue/context.py", line 233, in
 create_dynamic_frame_from_options\n    source.setFormat(format, **format_options)\n',
 info='')

 msg :  error in jackson reader
```

```
 stackTrace :  com.fasterxml.jackson.core.JsonParseException: Unexpected character
 ('{' (code 123)): was expecting either valid name character (for unquoted name) or
 double-quote (for quoted) to start field name
 at [Source: com.amazonaws.services.glue.readers.BufferedStream@73492578; line: 3,
 column: 2]
 at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1581)
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:533)
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportUnexpectedChar(ParserMinimalBase.java
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleOddName(UTF8StreamJsonParser.java:
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._parseName(UTF8StreamJsonParser.java:1650
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:740)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at scala.collection.Iterator$$anon$9.next(Iterator.scala:162)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:599)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:598)
 at scala.collection.Iterator$class.foreach(Iterator.scala:891)
 at scala.collection.AbstractIterator.foreach(Iterator.scala:1334)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:120)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:116)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErr(DynamicRecordBuilder.scala:209)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErrorWithException(DynamicRecordBuilder
 at
 com.amazonaws.services.glue.readers.JacksonReader.nextFailSafe(JacksonReader.scala:116)
 at com.amazonaws.services.glue.readers.JacksonReader.next(JacksonReader.scala:109)
 at com.amazonaws.services.glue.readers.JSONReader.next(JSONReader.scala:247)
 at
 com.amazonaws.services.glue.hadoop.TapeHadoopRecordReaderSplittable.nextKeyValue(TapeHadoopRec
 at org.apache.spark.rdd.NewHadoopRDD$$anon$1.hasNext(NewHadoopRDD.scala:230)
 at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:37)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
```

```
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:255)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:247)
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
 at org.apache.spark.scheduler.Task.run(Task.scala:121)
 at org.apache.spark.executor.Executor$TaskRunner$$anonfun$10.apply(Executor.scala:408)
 at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1360)
 at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:414)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
 at java.lang.Thread.run(Thread.java:750)


 input :

 bytesread :  252

 source :

 dynamicRecord :  Row(id=2, name='benjamin', surname='franklin')
```

errorsCount

`errorsCount( )` – Gibt die Gesamtzahl der Fehler in einem `DynamicFrame` zurück.

stageErrorsCount

`stageErrorsCount` – Gibt die Anzahl der aufgetretenen Fehler im Generierungsprozess dieses `DynamicFrame` zurück.

DynamicFrameCollection Klasse

Ein `DynamicFrameCollection` ist ein Verzeichnis von DynamicFrame Klasse-Objekten, in dem die Schlüssel die Namen der `DynamicFrames` und die Werte die `DynamicFrame` Objekte sind.

__init__

`__init__(dynamic_frames, glue_ctx)`

- `dynamic_frames` – Ein Verzeichnis von DynamicFrame Klasse-Objekten.
- `glue_ctx` – Ein GlueContext Klasse-Objekt.

Schlüssel

`keys( )` – Gibt eine Liste der Schlüssel in dieser Sammlung zurück, die in der Regel aus den Namen der entsprechenden `DynamicFrame`-Werte besteht.

Werte

`values(key)` – Gibt eine Liste der `DynamicFrame`-Werte in dieser Sammlung zurück.

Select

`select(key)`

Gibt das `DynamicFrame` zurück, das dem angegebenen Schlüssel entspricht (was in der Regel der Name des `DynamicFrame`) ist.

- `key` – Ein Schlüssel im `DynamicFrameCollection`, das in der Regel den Name eines `DynamicFrame` ist.

Zuordnung

`map(callable, transformation_ctx="")`

Verwendet eine übergebene Funktion, um auf der Grundlage von `DynamicFrames` in dieser Sammlung ein neue `DynamicFrameCollection` zu erstellen und zurückzugeben.

- `callable` – Eine Funktion, die ein `DynamicFrame` und den angegebenen Transformationskontext als Parameter entgegennimmt und ein `DynamicFrame` zurückgibt.

- `transformation_ctx` – Ein Transformationskontext, der vom Aufgerufenen verwendet werden soll (optional).

Flatmap

`flatmap(f, transformation_ctx="")`

Verwendet eine übergebene Funktion, um auf der Grundlage von `DynamicFrames` in dieser Sammlung ein neue `DynamicFrameCollection` zu erstellen und zurückzugeben.

- `f` – Eine Funktion, die ein `DynamicFrame` als Parameter entgegennimmt und ein `DynamicFrame` oder `DynamicFrameCollection` zurückgibt.
- `transformation_ctx` – Ein Transformationskontext, der von der Funktion verwendet werden soll (optional).

DynamicFrameWriter Klasse

Methoden

- __init__
- from_options
- from_catalog
- from_jdbc_conf

__init__

`__init__(glue_context)`

- `glue_context` – Der zu verwendende GlueContext Klasse.

from_options

`from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Schreibt einen `DynamicFrame` mit der angegebenen Verbindung und dem angegebenen Format.

- `frame` – Der zu schreibende `DynamicFrame`.

- `connection_type` – Der Verbindungstyp. Gültige Werte sind `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` und `oracle`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Für den `connection_type` `s3` ist ein Amazon-S3-Pfad definiert.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

> **Warning**
>
> Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Die `dbtable`-Eigenschaft ist der Name der JDBC-Tabelle. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie `schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet.

Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

- `format` – Eine Formatspezifikation (optional). Dies wird für einen Amazon Simple Storage Service (Amazon S3) verwendet oder AWS Glue Verbindung, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).

from_catalog

`from_catalog(frame, name_space, table_name, redshift_tmp_dir="", transformation_ctx="")`

Schreibt einen `DynamicFrame` mit der angegebenen Katalogdatenbank und dem angegebenen Tabellennamen.

- `frame` – Der zu schreibende `DynamicFrame`.
- `name_space` – Die zu verwendende Datenbank.
- `table_name` – Der zu verwendende `table_name`.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).
- `additional_options`— Zusätzliche Optionen stehen zur Verfügung AWS Glue.

  Um an zu schreiben Lake Formation Für verwaltete Tabellen können Sie die folgenden zusätzlichen Optionen verwenden:

  - `transactionId` – (String) Die Transaktions-ID, bei der das Schreiben in die Tabelle „Governed“ durchgeführt werden soll. Diese Transaktion kann nicht bereits festgeschrieben oder abgebrochen werden, sonst schlägt der Schreiben fehl.
  - `callDeleteObjectsOnCancel` — (Boolean, optional) Wenn auf `true` (Standard) gesetzt, AWS Glue ruft die `DeleteObjectsOnCancel` API automatisch auf, nachdem das Objekt in Amazon S3 geschrieben wurde. Weitere Informationen finden Sie unter DeleteObjectsOnCancel im AWS Lake Formation -Entwicklerhandbuch.

  Example Beispiel: Schreiben in eine verwaltete Tabelle in Lake Formation

```
txId = glueContext.start_transaction(read_only=False)
glueContext.write_dynamic_frame.from_catalog(
    frame=dyf,
    database = db,
    table_name = tbl,
    transformation_ctx = "datasource0",
    additional_options={"transactionId":txId})
...
glueContext.commit_transaction(txId)
```

from_jdbc_conf

`from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx="")`

Schreibt einen `DynamicFrame` mit den angegebenen JDBC-Verbindungsinformationen.

- `frame` – Der zu schreibende `DynamicFrame`.
- `catalog_connection` – Eine zu verwendende Katalogverbindung.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional).
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).

Beispiel für write_dynamic_frame

In diesem Beispiel wird die Ausgabe lokal mit einem `connection_type` von S3 mit einem POSIX-Pfadargument in `connection_options` geschrieben. Das ermöglicht das Schreiben in den lokalen Speicher.

```
glueContext.write_dynamic_frame.from_options(\
frame = dyf_splitFields,\
connection_options = {'path': '/home/glue/GlueLocalOutput/'},\
connection_type = 's3',\
format = 'json')
```

DynamicFrameReader Klasse

– Methoden –

- __init__
- from_rdd
- from_options
- from_catalog

__init__

`__init__(glue_context)`

- `glue_context` – Der zu verwendende GlueContext Klasse.

from_rdd

`from_rdd(data, name, schema=None, sampleRatio=None)`

Liest einen `DynamicFrame` aus einem Resilient Distributed Dataset (RDD).

- `data` – Das Dataset, aus dem gelesen werden soll.
- `name` – Der Name, der gelesen werden soll.
- `schema` – Das zu lesende Schema (optional).
- `sampleRatio` – Das Beispielverhältnis (optional).

from_options

`from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Liest einen `DynamicFrame` mit der angegebenen Verbindung und dem angegebenen Format.

- `connection_type` – Der Verbindungstyp. Zu den gültigen Werten gehören `s3`mysql,`postgresql`,`redshift`,`sqlserver`, `oracledynamodb`, und`snowflake`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue for Spark. Für einen `connection_type` `s3` werden Amazon-S3-Pfade in einem Array definiert.

  ```
  connection_options = {"paths": [ "s3://amzn-s3-demo-bucket/object_a", "s3://amzn-s3-demo-bucket/object_b"]}
  ```

  Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

> **Warning**
>
> Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Für eine JDBC-Verbindung, die paralleles Lesen durchführt, können Sie die Option Hashfield setzen. Zum Beispiel:

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path" , "hashfield": "month"}
```

Weitere Informationen finden Sie unter Parallel aus JDBC-Tabellen lesen.

- `format` – Eine Formatspezifikation (optional). Dies wird für einen Amazon Simple Storage Service (Amazon S3) verwendet oder AWS Glue Verbindung, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `push_down_predicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.

from_catalog

`from_catalog(database, table_name, redshift_tmp_dir="", transformation_ctx="", push_down_predicate="", additional_options={})`

Liest einen `DynamicFrame` mit dem angegebenen Katalog-Namespace und Tabellennamen.

- `database` – Die Datenbank, aus der gelesen werden soll.
- `table_name` –Der Name der Tabelle, aus der gelesen werden soll.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional, wenn keine Daten aus Redshift gelesen werden).
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `push_down_predicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.
- `additional_options`— Zusätzliche Optionen stehen zur Verfügung AWS Glue.
  - Um eine JDBC-Verbindung zu verwenden, die paralleles Lesen durchführt, können Sie die Optionen `hashfield`, `hashexpression` oder `hashpartitions` festlegen. Zum Beispiel:

    ```
    additional_options = {"hashfield": "month"}
    ```

    Weitere Informationen finden Sie unter Parallel aus JDBC-Tabellen lesen.
  - Sie können einen Katalogausdruck basierend auf den Indexspalten an den Filter übergeben. Sie sehen jetzt die Option `catalogPartitionPredicate`.

    `catalogPartitionPredicate` – Sie können einen Katalogausdruck basierend auf den Indexspalten an den Filter übergeben. Dies verlagert die Filterung auf die Serverseite. Weitere Informationen finden Sie unter .AWS Glue Partitionsindizes. Beachten Sie, dass `push_down_predicate` und `catalogPartitionPredicate` verschiedene Syntaxen verwenden. Erstere verwendet die Spark-SQL-Standardsyntax und letztere verwendet den JSQL-Parser.

    Weitere Informationen finden Sie unter Partitionen für die ETL-Ausgabe verwalten in AWS Glue.

GlueContext Klasse

Umschließt das Apache SparkContextSpark-Objekt und bietet dadurch Mechanismen für die Interaktion mit der Apache Spark-Plattform.

__init__

`__init__(sparkContext)`

- `sparkContext` – Der Apache-Spark-Kontext, der verwendet werden soll.

Erstellen

- __init__
- getSource
- create_dynamic_frame_from_rdd
- create_dynamic_frame_from_catalog
- create_dynamic_frame_from_options
- create_sample_dynamic_frame_from_catalog
- create_sample_dynamic_frame_from_options
- add_ingestion_time_columns
- create_data_frame_from_catalog
- create_data_frame_from_options
- forEachBatch

getSource

`getSource(connection_type, transformation_ctx = "", **options)`

Erstellt ein `DataSource`-Objekt, das zum Lesen von `DynamicFrames` aus externen Quellen verwendet werden kann.

- `connection_type` – Der zu verwendende Verbindungstyp, z. B. Amazon Simple Storage Service (Amazon S3), Amazon Redshift und JDBC. Gültige Werte sind unter anderem `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle` und `dynamodb`.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `options` – Eine Sammlung optionaler Name/Wert-Paare. Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

Es folgt ein Beispiel für die Verwendung von `getSource`.

```
>>> data_source = context.getSource("file", paths=["/in/path"])
>>> data_source.setFormat("json")
>>> myFrame = data_source.getFrame()
```

create_dynamic_frame_from_rdd

`create_dynamic_frame_from_rdd(data, name, schema=None, sample_ratio=None, transformation_ctx="")`

Gibt einen `DynamicFrame` zurück, der aus einem Apache Spark Resilient Distributed Dataset (RDD) erstellt wird.

- `data` – Die Datenquelle, die verwendet werden soll.
- `name` – Der Name der zu verwendenden Daten.
- `schema` – Das zu verwendende Schema (optional).
- `sample_ratio` – Das zu verwendende Beispielverhältnis.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).

create_dynamic_frame_from_catalog

`create_dynamic_frame_from_catalog(database, table_name, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, catalog_id = None)`

Gibt einen `DynamicFrame` zurück, der mit einer Data-Catalog-Datenbank und einem Tabellennamen erstellt wird. Wenn Sie diese Methode verwenden, geben Sie `format_options` über die Tabelle Eigenschaften der angegebenen AWS Glue-Datenkatalogtabelle und andere Optionen über das `additional_options` Argument an.

- `Database` – Die Datenbank, aus der gelesen werden soll.
- `table_name` –Der Name der Tabelle, aus der gelesen werden soll.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `push_down_predicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Informationen zu unterstützten Quellen und Einschränkungen finden Sie unter Optimieren von Lesevorgängen mit Pushdown in AWS Glue ETL. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.
- `additional_options` – Eine Sammlung optionaler Name/Wert-Paare. Zu den möglichen Optionen gehören die unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark aufgeführten, außer `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`,

`topicName`, `classification` und `delimiter`. Eine weitere unterstützte Option ist `catalogPartitionPredicate`:

`catalogPartitionPredicate` – Sie können einen Katalogausdruck basierend auf den Indexspalten an den Filter übergeben. Dies verlagert die Filterung auf die Serverseite. Weitere Informationen finden Sie unter AWS Glue-Partition-Indizes. Beachten Sie, dass `push_down_predicate` und `catalogPartitionPredicate` verschiedene Syntaxen verwenden. Erstere verwendet die Spark-SQL-Standardsyntax und letztere verwendet den JSQL-Parser.

- `catalog_id` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei None wird die Standard-Konto-ID des Aufrufers verwendet.

create_dynamic_frame_from_options

`create_dynamic_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Gibt einen `DynamicFrame` zurück, der mit der angegebenen Verbindung und dem Format erstellt wurde.

- `connection_type` – Der Verbindungstyp, z. B. Amazon S3, Amazon Redshift und JDBC. Gültige Werte sind unter anderem `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle` und `dynamodb`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfade und Datenbanktabelle (optional). Für einen `connection_type` von `s3` ist eine Liste der Amazon-S3-Pfade definiert.

```
connection_options = {"paths": ["s3://aws-glue-target/temp"]}
```

Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

> **Warning**
>
> Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Die `dbtable`-Eigenschaft ist der Name der JDBC-Tabelle. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie `schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet.

Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

- `format`— Eine Formatspezifikation. Diese wird für eine Amazon-S3- oder eine AWS Glue-Verbindung verwendet, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `push_down_predicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Informationen zu unterstützten Quellen und Einschränkungen finden Sie unter Optimieren von Lesevorgängen mit Pushdown in AWS Glue ETL. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.

create_sample_dynamic_frame_from_catalog

`create_sample_dynamic_frame_from_catalog(database, table_name, num, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, sample_options = {}, catalog_id = None)`

Gibt einen Beispiel-`DynamicFrame` zurück, der mit einer Data-Catalog-Datenbank und einem Tabellennamen erstellt wird. `DynamicFrame` enthält nur erste `num`-Datensätze aus einer Datenquelle.

- `database` – Die Datenbank, aus der gelesen werden soll.
- `table_name` –Der Name der Tabelle, aus der gelesen werden soll.

- num – Die maximale Anzahl von Datensätzen im dynamischen Frame der zurückgegebenen Probe.
- redshift_tmp_dir – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- transformation_ctx – Der zu verwendende Transformationskontext (optional).
- push_down_predicate – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.
- additional_options – Eine Sammlung optionaler Name/Wert-Paare. Zu den möglichen Optionen gehören die unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark aufgeführten, außer endpointUrl, streamName, bootstrap.servers, security.protocol, topicName, classification und delimiter.
- sample_options – Parameter zur Steuerung des Sampling-Verhaltens (optional). Aktuell verfügbare Parameter für Amazon-S3-Quellen:
  - maxSamplePartitions – Die maximale Anzahl der Partitionen, die das Sampling lesen wird. Standardwert: 10.
  - maxSampleFilesPerPartition – Die maximale Anzahl von Dateien, die das Sampling in einer Partition lesen wird. Standardwert: 10.

    Diese Parameter helfen, den Zeitaufwand für die Dateiauflistung zu reduzieren. Angenommen, der Datensatz hat 1 000 Partitionen und jede Partition hat 10 Dateien. Wenn Sie maxSamplePartitions = 10 und maxSampleFilesPerPartition = 10 festlegen, listet das Sampling statt alle 10 000 Dateien aufzulisten und liest nur die ersten 10 Partitionen mit jeweils den ersten 10 Dateien: 10*10 = 100 Dateien insgesamt.
- catalog_id – Die Katalog-ID des Data Catalogs, auf den zugegriffen wird (die Konto-ID des Data Catalogs). Standardmäßig auf None eingestellt. None ist standardmäßig die Katalog-ID des aufrufenden Kontos im Service.

create_sample_dynamic_frame_from_options

```
create_sample_dynamic_frame_from_options(connection_type,
connection_options={}, num, sample_options={}, format=None,
format_options={}, transformation_ctx = "")
```

Gibt einen Beispiel-DynamicFrame zurück, der mit der angegebenen Verbindung und dem Format erstellt wurde. DynamicFrame enthält nur erste num-Datensätze aus einer Datenquelle.

- `connection_type` – Der Verbindungstyp, z. B. Amazon S3, Amazon Redshift und JDBC. Gültige Werte sind unter anderem `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle` und `dynamodb`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfade und Datenbanktabelle (optional). Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.
- `num` – Die maximale Anzahl von Datensätzen im dynamischen Frame der zurückgegebenen Probe.
- `sample_options` – Parameter zur Steuerung des Sampling-Verhaltens (optional). Aktuell verfügbare Parameter für Amazon-S3-Quellen:
  - `maxSamplePartitions` – Die maximale Anzahl der Partitionen, die das Sampling lesen wird. Standardwert: 10.
  - `maxSampleFilesPerPartition` – Die maximale Anzahl von Dateien, die das Sampling in einer Partition lesen wird. Standardwert: 10.

    Diese Parameter helfen, den Zeitaufwand für die Dateiauflistung zu reduzieren. Angenommen, der Datensatz hat 1 000 Partitionen und jede Partition hat 10 Dateien. Wenn Sie `maxSamplePartitions` = 10 und `maxSampleFilesPerPartition` = 10 festlegen, listet das Sampling statt alle 10 000 Dateien aufzulisten und liest nur die ersten 10 Partitionen mit jeweils den ersten 10 Dateien: 10*10 = 100 Dateien insgesamt.
- `format`— Eine Formatspezifikation. Diese wird für eine Amazon-S3- oder eine AWS Glue-Verbindung verwendet, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `push_down_predicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.

add_ingestion_time_columns

`add_ingestion_time_columns(dataFrame, timeGranularity = "")`

Hängt die Erfassungszeitspalten wie `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` an die Eingabe `DataFrame` an. Diese Funktion wird automatisch in dem von AWS Glue erzeugten Skript generiert, wenn Sie eine Data-Catalog-Tabelle mit Amazon S3 als Ziel angeben. Diese Funktion aktualisiert automatisch die Partition mit Erfassungszeitspalten in der Ausgabetabelle. So können die Ausgabedaten bei der Erfassung automatisch partitioniert werden, ohne dass explizite Erfassungszeitspalten in den Eingabedaten erforderlich sind.

- `dataFrame` – Der `dataFrame`, um die Erfassungszeitspalten anzuhängen.
- `timeGranularity` – Die Granularität der Zeitspalten. Gültige Werte sind „`day`“, „`hour`“ und „`minute`“. Wenn zum Beispiel „`hour`“ an die Funktion übergeben wird, werden im Original `dataFrame` die Zeiten in den Spalten „`ingest_year`“, „`ingest_month`“, „`ingest_day`“, und „`ingest_hour`“ aktualisiert.

Gibt den Datenrahmen nach dem Anhängen der Zeitgranularitätsspalten zurück.

Beispiel:

```
dynamic_frame = DynamicFrame.fromDF(glueContext.add_ingestion_time_columns(dataFrame,
 "hour"))
```

create_data_frame_from_catalog

`create_data_frame_from_catalog(database, table_name, transformation_ctx = "", additional_options = {})`

Gibt einen `DataFrame` zurück, der mit Informationen aus einer Data-Catalog-Tabelle erstellt wird.

- `database` – Die Data-Catalog-Datenbank, aus der gelesen werden soll.
- `table_name` – Der Name der Data-Catalog-Tabelle, aus der gelesen werden soll.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `additional_options` – Eine Sammlung optionaler Name/Wert-Paare. Zu den möglichen Optionen gehören die unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark für Streaming-Quellen aufgelisteten wie `startingPosition`, `maxFetchTimeInMs` und `startingOffsets`.

- `useSparkDataSource`— Wenn auf true gesetzt, zwingt AWS Glue, die native Spark-Datenquellen-API zum Lesen der Tabelle zu verwenden. Die Spark-Datenquellen-API unterstützt die folgenden Formate: AVRO, Binär, CSV, JSON, ORC, Parquet und Text. In einer Data-Catalog-Tabelle geben Sie das Format mithilfe der `classification`-Eigenschaft an. Weitere Informationen zur Spark-Datenquellen-API finden Sie in der offiziellen Apache-Spark-Dokumentation.

  Die Verwendung von `create_data_frame_from_catalog` mit `useSparkDataSource` bietet die folgenden Vorteile:

  - Gibt direkt ein `DataFrame` zurück und bietet eine Alternative zu `create_dynamic_frame.from_catalog().toDF()`.
  - Unterstützt die Berechtigungssteuerung AWS Lake Formation auf Tabellenebene für native Formate.
  - Unterstützt das Lesen von Data-Lake-Formaten ohne Berechtigungssteuerung auf AWS Lake Formation Tabellenebene. Weitere Informationen finden Sie unter Verwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

  Wenn Sie die Option aktivieren`useSparkDataSource`, können Sie bei Bedarf auch jede der Spark-Datenquellenoptionen hinzufügen. `additional_options` AWS Glue leitet diese Optionen direkt an das Spark-Lesegerät weiter.
- `useCatalogSchema`— Wenn auf true gesetzt, wendet AWS Glue das Datenkatalogschema auf das Ergebnis an`DataFrame`. Andernfalls leitet der Reader das Schema aus den Daten ab. Wenn Sie `useCatalogSchema` aktivieren, müssen Sie auch `useSparkDataSource` auf wahr setzen.

Einschränkungen

Berücksichtigen Sie die folgenden Einschränkungen, wenn Sie die `useSparkDataSource`-Option verwenden:

- Wenn Sie AWS Glue verwenden`useSparkDataSource`, erstellt es `DataFrame` in einer separaten Spark-Sitzung eine neue, die sich von der ursprünglichen Spark-Sitzung unterscheidet.
- Die DataFrame Spark-Partitionsfilterung funktioniert nicht mit den folgenden AWS Glue-Funktionen.
  - Auftrags-Lesezeichen
  - Ausschließen von Amazon-S3-Speicherklassen
  - Katalogpartitionsprädikate

Um die Partitionsfilterung mit diesen Funktionen zu verwenden, können Sie das AWS Glue-Pushdown-Prädikat verwenden. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten. Die Filterung nach nicht partitionierten Spalten ist nicht betroffen.

Das folgende Beispielskript zeigt die falsche Methode zur Durchführung der Partitionsfilterung mit der `excludeStorageClasses`-Option.

```
// Incorrect partition filtering using Spark filter with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)

//  Suppose year and month are partition keys.
//  Filtering on year and month won't work, the filtered_df will still
//  contain data with other year/month values.
filtered_df = read_df.filter("year == '2017 and month == '04' and 'state == 'CA'")
```

Das folgende Beispielskript zeigt die korrekte Verwendung eines Pushdown-Prädikats, um eine Partitionsfilterung mit der `excludeStorageClasses`-Option durchzuführen.

```
// Correct partition filtering using the AWS Glue pushdown predicate
// with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    //  Use AWS Glue pushdown predicate to perform partition filtering
    push_down_predicate = "(year=='2017' and month=='04')"
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)

//  Use Spark filter only on non-partitioned columns
filtered_df = read_df.filter("state == 'CA'")
```

Beispiel: Erstellen einer CSV-Tabelle mit dem Spark-Datenquellen-Reader

```
//  Read a CSV table with '\t' as separator
read_df = glueContext.create_data_frame.from_catalog(
    database=<database_name>,
    table_name=<table_name>,
    additional_options = {"useSparkDataSource": True,  "sep": '\t'}
)
```

create_data_frame_from_options

`create_data_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Diese API ist jetzt veraltet. Verwenden Sie stattdessen die `getSource()`-APIs. Gibt einen `DataFrame` zurück, der mit der angegebenen Verbindung und dem Format erstellt wurde. Verwenden Sie diese Funktion nur mit AWS Glue-Streaming-Quellen.

- `connection_type` – Der Streaming-Verbindungstyp. Gültige Werte sind `kinesis` und `kafka`.
- `connection_options` – Verbindungsoptionen, die für Kinesis und Kafka unterschiedlich sind. Die Liste aller Verbindungsoptionen für jede Streaming-Datenquelle finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark. Beachten Sie die folgenden Unterschiede bei den Streaming-Verbindungsoptionen:
  - Kinesis-Streaming-Quellen erfordern `streamARN`, `startingPosition`, `inferSchema` und `classification`.
  - Kafka-Streaming-Quellen erfordern `connectionName`, `topicName`, `startingOffsets`, `inferSchema` und `classification`.
- `format`— Eine Formatspezifikation. Diese wird für eine Amazon-S3- oder eine AWS Glue-Verbindung verwendet, die mehrere Formate unterstützt. Weitere Informationen zu unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Weitere Informationen zu unterstützten Formatoptionen finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Beispiel für die Kafka-Streaming-Quelle:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

Wendet die `batch_function` auf jeden Mikrobatch an, der von der Streaming-Quelle gelesen wird.

- `frame`— Der DataFrame , der die aktuelle Mikrocharge enthält.
- `batch_function` – Eine Funktion, die für jeden Mikrobatch angewendet wird.
- `options` – Eine Sammlung von Schlüssel-Wert-Paaren, die Informationen zur Verarbeitung von Mikrobatches enthält. Die folgenden Optionen sind erforderlich:
  - `windowSize` – Die Zeitspanne für die Verarbeitung der einzelnen Batches.
  - `checkpointLocation` – Der Ort, an dem Checkpoints für den Streaming-ETL-Auftrag gespeichert werden.
  - `batchMaxRetries` – Die maximale Anzahl der Wiederholungsversuche für diesen Batch, wenn er fehlschlägt. Der Standardwert ist 3. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.

Beispiel:

```
glueContext.forEachBatch(
    frame = data_frame_datasource0,
    batch_function = processBatch,
    options = {
        "windowSize": "100 seconds",
        "checkpointLocation": "s3://kafka-auth-dataplane/confluent-test/output/
checkpoint/"
    }
)

def processBatch(data_frame, batchId):
    if (data_frame.count() > 0):
        datasource0 = DynamicFrame.fromDF(
          glueContext.add_ingestion_time_columns(data_frame, "hour"),
          glueContext, "from_data_frame"
        )
        additionalOptions_datasink1 = {"enableUpdateCatalog": True}
        additionalOptions_datasink1["partitionKeys"] = ["ingest_yr", "ingest_mo",
 "ingest_day"]
        datasink1 = glueContext.write_dynamic_frame.from_catalog(
          frame = datasource0,
          database = "tempdb",
          table_name = "kafka-auth-table-output",
          transformation_ctx = "datasink1",
          additional_options = additionalOptions_datasink1
        )
```

Arbeiten mit Datensätzen in Amazon S3

- purge_table
- purge_s3_path
- transition_table
- transition_s3_path

purge_table

`purge_table(catalog_id=None, database="", table_name="", options={}, transformation_ctx="")`

Löscht Dateien aus Amazon S3 für die Datenbank und Tabelle des angegebenen Katalogs. Wenn alle Dateien in einer Partition gelöscht werden, wird diese Partition auch aus dem Katalog gelöscht. Wir unterstützen keine purge_table-Aktion für Tabellen, die bei Lake Formation registriert sind.

Wenn Sie gelöschte Objekte wiederherstellen möchten, können Sie das Objekt-Versioning auf dem Amazon-S3-Bucket aktivieren. Wenn ein Objekt aus einem Bucket gelöscht wird, für den kein Objekt-Versioning aktiviert ist, kann das Objekt nicht wiederhergestellt werden. Weitere Informationen zum Wiederherstellen gelöschter Objekte in einem versionsfähigen Bucket finden Sie unter How can I retrieve an Amazon S3 object that was deleted? (Wie kann ich ein gelöschtes Amazon-S3-Objekt abrufen? im AWS -Support Wissenscenter.

- `catalog_id` – Die Katalog-ID des Data Catalogs, auf den zugegriffen wird (die Konto-ID des Data Catalogs). Standardmäßig auf `None` eingestellt. `None` ist standardmäßig die Katalog-ID des aufrufenden Kontos im Service.
- `database` – Die zu verwendende Datenbank.
- `table_name` – Der Name der zu verwendenden Tabelle.
- `options` – Optionen zum Filtern von zu löschenden Dateien und zur Generierung von Manifestdateien.
  - `retentionPeriod` – Gibt einen Zeitraum in Stunden für die Beibehaltung von Dateien an. Dateien, die älter als der Aufbewahrungszeitraum sind, werden beibehalten. Standardmäßig auf 168 Stunden (7 Tage) eingestellt.
  - `partitionPredicate` – Partitionen, die diese Bedingung erfüllen, werden gelöscht. Dateien innerhalb des Aufbewahrungszeitraums in diesen Partitionen werden nicht gelöscht. Festgelegt auf `""` – standardmäßig auf leer festgelegt.
  - `excludeStorageClasses` – Dateien mit Speicherklasse im `excludeStorageClasses`-Set werden nicht gelöscht. Der Standardwert ist `Set()` – ein leeres Set.
  - `manifestFilePath` – Ein optionaler Pfad für die Generierung von Manifestdateien. Alle Dateien, die erfolgreich gelöscht wurden, werden in `Success.csv` aufgezeichnet und diejenigen, für die dies fehlgeschlagen ist, in `Failed.csv`
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional). Wird im Dateipfad des Manifests verwendet.

Example

```
glueContext.purge_table("database", "table", {"partitionPredicate": "(month=='march')",
 "retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath":
 "s3://bucketmanifest/"})
```

purge_s3_path

`purge_s3_path(s3_path, options={}, transformation_ctx="")`

Löscht Dateien rekursiv aus dem angegebenen Amazon-S3-Pfad.

Wenn Sie gelöschte Objekte wiederherstellen möchten, können Sie das Objekt-Versioning auf dem Amazon-S3-Bucket aktivieren. Wenn ein Objekt aus einem Bucket gelöscht wird, für den kein Objekt-Versioning aktiviert ist, kann das Objekt nicht wiederhergestellt werden. Weitere Informationen zum Wiederherstellen gelöschter Objekte in einem Bucket mit Versionierung finden Sie unter Wie kann ich ein gelöschtes Amazon S3 S3-Objekt abrufen? im Support Knowledge Center.

- `s3_path` – Der Pfad in Amazon S3 der zu löschenden Dateien im Format `s3://<bucket>/<prefix>/`.
- `options` – Optionen zum Filtern von zu löschenden Dateien und zur Generierung von Manifestdateien.
  - `retentionPeriod` – Gibt einen Zeitraum in Stunden für die Beibehaltung von Dateien an. Dateien, die älter als der Aufbewahrungszeitraum sind, werden beibehalten. Standardmäßig auf 168 Stunden (7 Tage) eingestellt.
  - `excludeStorageClasses` – Dateien mit Speicherklasse im `excludeStorageClasses`-Set werden nicht gelöscht. Der Standardwert ist `Set()` – ein leeres Set.
  - `manifestFilePath` – Ein optionaler Pfad für die Generierung von Manifestdateien. Alle Dateien, die erfolgreich gelöscht wurden, werden in `Success.csv` aufgezeichnet und diejenigen, für die dies fehlgeschlagen ist, in `Failed.csv`
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional). Wird im Dateipfad des Manifests verwendet.

Example

```
glueContext.purge_s3_path("s3://bucket/path/", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/"})
```

transition_table

`transition_table(database, table_name, transition_to, options={}, transformation_ctx="", catalog_id=None)`

Wandelt die Speicherklasse der Dateien, die in Amazon S3 für die Datenbank und die Tabelle des angegebenen Katalogs gespeichert sind um.

Sie können zwischen zwei beliebigen Speicherklassen wechseln. Für die Speicherklassen `GLACIER` und `DEEP_ARCHIVE` können Sie zu diesen Klassen wechseln. Sie würden jedoch eine `S3 RESTORE` für den Übergang von den Speicherklassen `GLACIER` und `DEEP_ARCHIVE` verwenden.

Wenn Sie AWS Glue-ETL-Aufträge ausführen, die Dateien oder Partitionen aus Amazon S3 lesen, können Sie einige Amazon-S3-Speicherklassentypen ausschließen. Weitere Informationen finden Sie unter Ausschließen von Amazon-S3-Speicherklassen.

- `database` – Die zu verwendende Datenbank.
- `table_name` – Der Name der zu verwendenden Tabelle.
- `transition_to` – Die Amazon-S3-Speicherklassen zum Umwandeln.
- `options` – Optionen zum Filtern von zu löschenden Dateien und zur Generierung von Manifestdateien.
  - `retentionPeriod` – Gibt einen Zeitraum in Stunden für die Beibehaltung von Dateien an. Dateien, die älter als der Aufbewahrungszeitraum sind, werden beibehalten. Standardmäßig auf 168 Stunden (7 Tage) eingestellt.
  - `partitionPredicate` – Partitionen, die diese Bedingung erfüllen, werden umgewandelt. Dateien innerhalb des Aufbewahrungszeitraums in diesen Partitionen werden nicht transitioniert. Festgelegt auf `""` – standardmäßig auf leer festgelegt.
  - `excludeStorageClasses` – Dateien mit Speicherklasse im `excludeStorageClasses`-Set werden nicht umgewandelt. Der Standardwert ist `Set()` – ein leeres Set.
  - `manifestFilePath` – Ein optionaler Pfad für die Generierung von Manifestdateien. Alle Dateien, die erfolgreich transitioniert wurden, werden in `Success.csv` aufgezeichnet und diejenigen, für die dies fehlgeschlagen ist, in `Failed.csv`
  - `accountId` – Die Konto-ID von Amazon Web Services zum Ausführen der Umwandlungstransformation. Obligatorisch für diese Transformation.
  - `roleArn`— Die AWS Rolle bei der Durchführung der Transformationstransformation. Obligatorisch für diese Transformation.

- `transformation_ctx` – Der zu verwendende Transformationskontext (optional). Wird im Dateipfad des Manifests verwendet.
- `catalog_id` – Die Katalog-ID des Data Catalogs, auf den zugegriffen wird (die Konto-ID des Data Catalogs). Standardmäßig auf None eingestellt. None ist standardmäßig die Katalog-ID des aufrufenden Kontos im Service.

Example

```
glueContext.transition_table("database", "table", "STANDARD_IA", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/",
 "accountId": "12345678901", "roleArn": "arn:aws:iam::123456789012:user/example-
username"})
```

transition_s3_path

`transition_s3_path(s3_path, transition_to, options={}, transformation_ctx="")`

Rekursive Umwandlung der Speicherklasse der Dateien im angegebenen Amazon-S3-Pfad.

Sie können zwischen zwei beliebigen Speicherklassen wechseln. Für die Speicherklassen `GLACIER` und `DEEP_ARCHIVE` können Sie zu diesen Klassen wechseln. Sie würden jedoch eine `S3 RESTORE` für den Übergang von den Speicherklassen `GLACIER` und `DEEP_ARCHIVE` verwenden.

Wenn Sie AWS Glue-ETL-Aufträge ausführen, die Dateien oder Partitionen aus Amazon S3 lesen, können Sie einige Amazon-S3-Speicherklassentypen ausschließen. Weitere Informationen finden Sie unter Ausschließen von Amazon-S3-Speicherklassen.

- `s3_path` – Der Pfad in Amazon S3 der umzuwandelnden Dateien im Format `s3://<bucket>/<prefix>/`.
- `transition_to` – Die Amazon-S3-Speicherklassen zum Umwandeln.
- `options` – Optionen zum Filtern von zu löschenden Dateien und zur Generierung von Manifestdateien.
  - `retentionPeriod` – Gibt einen Zeitraum in Stunden für die Beibehaltung von Dateien an. Dateien, die älter als der Aufbewahrungszeitraum sind, werden beibehalten. Standardmäßig auf 168 Stunden (7 Tage) eingestellt.

- `partitionPredicate` – Partitionen, die diese Bedingung erfüllen, werden umgewandelt. Dateien innerhalb des Aufbewahrungszeitraums in diesen Partitionen werden nicht transitioniert. Festgelegt auf `""` – standardmäßig auf leer festgelegt.
- `excludeStorageClasses` – Dateien mit Speicherklasse im `excludeStorageClasses`-Set werden nicht umgewandelt. Der Standardwert ist `Set()` – ein leeres Set.
- `manifestFilePath` – Ein optionaler Pfad für die Generierung von Manifestdateien. Alle Dateien, die erfolgreich transitioniert wurden, werden in `Success.csv` aufgezeichnet und diejenigen, für die dies fehlgeschlagen ist, in `Failed.csv`
- `accountId` – Die Konto-ID von Amazon Web Services zum Ausführen der Umwandlungstransformation. Obligatorisch für diese Transformation.
- `roleArn`— Die AWS Rolle zur Durchführung der Übergangstransformation. Obligatorisch für diese Transformation.

- `transformation_ctx` – Der zu verwendende Transformationskontext (optional). Wird im Dateipfad des Manifests verwendet.

Example

```
glueContext.transition_s3_path("s3://bucket/prefix/", "STANDARD_IA",
 {"retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"],
 "manifestFilePath": "s3://bucketmanifest/", "accountId": "12345678901", "roleArn":
 "arn:aws:iam::123456789012:user/example-username"})
```

Extrahieren

- extract_jdbc_conf

extract_jdbc_conf

`extract_jdbc_conf(connection_name, catalog_id = None)`

Gibt einen `dict` mit Schlüsseln zurück mit den Konfigurationseigenschaften aus dem AWS Glue-Verbindungsobjekt im Datenkatalog.

- `user` – Der Datenbankbenutzername.
- `password` – Das Datenbankpasswort.
- `vendor` – Gibt einen Anbieter an (`mysql`, `postgresql`, `oracle`, `sqlserver`, usw.).

- `enforceSSL` – Eine boolesche Zeichenfolge, die angibt, ob eine sichere Verbindung erforderlich ist.
- `customJDBCCert` – Verwenden Sie ein bestimmtes Client-Zertifikat aus dem angegebenen Amazon-S3-Pfad.
- `skipCustomJDBCCertValidation` – Eine boolesche Zeichenfolge, die angibt, ob das `customJDBCCert` von einer CA validiert werden muss.
- `customJDBCCertString` – Zusätzliche Informationen zum benutzerdefinierten Zertifikat, spezifisch für den Treibertyp.
- `url` – (Veraltet) JDBC-URL nur mit Protokoll, Server und Port.
- `fullUrl` – JDBC-URL wie beim Erstellen der Verbindung eingegeben (Verfügbar in AWS Glue-Version 3.0 oder höher).

Beispiel zum Abrufen von JDBC-Konfigurationen:

```
jdbc_conf = glueContext.extract_jdbc_conf(connection_name="your_glue_connection_name")
print(jdbc_conf)
>>> {'enforceSSL': 'false', 'skipCustomJDBCCertValidation': 'false', 'url':
 'jdbc:mysql://myserver:3306', 'fullUrl': 'jdbc:mysql://myserver:3306/mydb',
 'customJDBCCertString': '', 'user': 'admin', 'customJDBCCert': '', 'password': '1234',
 'vendor': 'mysql'}
```

Transaktionen

- start_transaction
- commit_transaction
- cancel_transaction

start_transaction

`start_transaction(read_only)`

Starten Sie eine neue Transaktion. Ruft intern die startTransaction-API von Lake Formation auf.

- `read_only` – (Boolean) Gibt an, ob diese Transaktion schreibgeschützt oder gelesen und geschrieben werden soll. Schreibvorgänge mit einer schreibgeschützten Transaktions-ID werden abgelehnt. Schreibgeschützte Transaktionen müssen nicht festgeschrieben werden.

Gibt die Transaktions-ID zurück.

commit_transaction

`commit_transaction(transaction_id, wait_for_commit = True)`

Versucht, die angegebene Transaktion zu übernehmen. `commit_transaction` kann zurückkehren, bevor die Transaktion abgeschlossen ist. Ruft intern die CommitTransaction-API von Lake Formation auf.

- `transaction_id` – (String) Die zu verbindende Transaktion.
- `wait_for_commit` – (Boolean) Bestimmt, ob die Rückgabe von `commit_transaction` sofort erfolgt. Der Standardwert ist "True". Wenn false (falsch), befragt `commit_transaction` und wartet, bis die Transaktion übergeben wurde. Die Dauer der Wartezeit ist mit einem exponentiellen Backoff mit maximal 6 Wiederholungsversuchen auf 1 Minute beschränkt.

Gibt einen booleschen Wert zurück, um anzugeben, ob das Commit abgeschlossen ist oder nicht.

cancel_transaction

`cancel_transaction(transaction_id)`

Versucht, die angegebene Transaktion abzubrechen. Gibt eine `TransactionCommittedException`-Ausnahme zurück, wenn die Transaktion zuvor festgeschrieben wurde. Ruft intern die Lake Formation CancelTransactionAPI auf.

- `transaction_id` – (String) Die abzubrechende Transaktion.

Writing

- getSink
- write_dynamic_frame_from_options
- write_from_options
- write_dynamic_frame_from_catalog
- write_data_frame_from_catalog
- write_dynamic_frame_from_jdbc_conf
- write_from_jdbc_conf

getSink

`getSink(connection_type, format = None, transformation_ctx = "", **options)`

Ruft ein `DataSink`-Objekt ab, das zum Schreiben von `DynamicFrames` in externen Quellen verwendet werden kann. Prüfen Sie zunächst das SparkSQL-`format`, um sicherzustellen, dass Sie die erwartete Senke erhalten.

- `connection_type` – Der Verbindungstyp, z. B. Amazon S3, Amazon Redshift und JDBC. Zu den gültigen Werten gehören `s3` `mysql`,`postgresql`,`redshift`,`sqlserver`,`oracle`,`kinesis`, und`kafka`.
- `format` – Das zu verwendende SparkSQL-Format (optional).
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `options` – Eine Sammlung von Name/Wert-Paaren, die zur Angabe der Verbindungsoptionen verwendet werden. Einige der möglichen Werte sind:
  - `user` und `password`: Zur Autorisierung
  - `url`: Der Endpunkt für den Datenspeicher
  - `dbtable`: Der Name der Zieltabelle
  - `bulkSize`: Grad der Parallelität für Insert-Operationen

Die Optionen, die Sie angeben können, hängen vom Verbindungstyp ab. In Verbindungsarten und Optionen für ETL in AWS Glue für Spark finden Sie zusätzliche Werte und Beispiele.

Beispiel:

```
>>> data_sink = context.getSink("s3")
>>> data_sink.setFormat("json"),
>>> data_sink.writeFrame(myFrame)
```

write_dynamic_frame_from_options

`write_dynamic_frame_from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Schreibt und gibt einen `DynamicFrame` mit der angegebenen Verbindung und dem angegebenen Format zurück.

- `frame` – Der zu schreibende `DynamicFrame`.
- `connection_type` – Der Verbindungstyp, z. B. Amazon S3, Amazon Redshift und JDBC. Zu den gültigen Werten gehören `s3` `mysqlpostgresql`,`redshift`,`sqlserver`,`oracle`,`kinesis`, und`kafka`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Für den `connection_type` `s3` ist ein Amazon-S3-Pfad definiert.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

> **Warning**
>
> Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Die `dbtable`-Eigenschaft ist der Name der JDBC-Tabelle. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie `schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet.

Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

- `format`— Eine Formatspezifikation. Diese wird für eine Amazon-S3- oder eine AWS Glue-Verbindung verwendet, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.

- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).

write_from_options

`write_from_options(frame_or_dfc, connection_type, connection_options={}, format={}, format_options={}, transformation_ctx = "")`

Schreibt und gibt einen `DynamicFrame` oder eine `DynamicFrameCollection` zurück, der bzw. die mit den angegebenen Verbindungs- und Formatinformationen erstellt wird.

- `frame_or_dfc` – Der `DynamicFrame` oder die `DynamicFrameCollection`, der bzw. die geschrieben werden soll.
- `connection_type` – Der Verbindungstyp, z. B. Amazon S3, Amazon Redshift und JDBC. Gültige Werte sind `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` und `oracle`.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Für den `connection_type` `s3` ist ein Amazon-S3-Pfad definiert.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Für JDBC-Verbindungen müssen mehrere Eigenschaften definiert werden. Beachten Sie, dass der Datenbankname Teil der URL sein muss. Er kann optional in die Verbindungsoptionen eingeschlossen werden.

> **Warning**
>
> Das Speichern von Passwörtern in Ihrem Skript wird nicht empfohlen. Erwägen Sie, sie `boto3` zu verwenden, um sie aus AWS Secrets Manager dem AWS Glue-Datenkatalog abzurufen.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

Die `dbtable`-Eigenschaft ist der Name der JDBC-Tabelle. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie `schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet.

Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.

- `format`— Eine Formatspezifikation. Diese wird für eine Amazon-S3- oder eine AWS Glue-Verbindung verwendet, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `format_options` – Formatoptionen für das angegebene Format. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).

write_dynamic_frame_from_catalog

`write_dynamic_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Schreibt und gibt einen `DynamicFrame` zurück, der mit einer Data-Catalog-Datenbank und einem Tabellennamen erstellt wird.

- `frame` – Der zu schreibende `DynamicFrame`.
- `Database` – Die Data-Catalog-Datenbank, die die Tabelle enthält.
- `table_name` – Der Name der Data-Catalog-Tabelle, die dem Ziel zugeordnet ist.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `additional_options` – Eine Sammlung optionaler Name/Wert-Paare.
- `catalog_id` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei None wird die Standard-Konto-ID des Aufrufers verwendet.

write_data_frame_from_catalog

`write_data_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Schreibt und gibt einen `DataFrame` zurück, der mit einer Data-Catalog-Datenbank und einem Tabellennamen erstellt wird. Diese Methode unterstützt das Schreiben in Data-Lake-Formate (Hudi, Iceberg und Delta Lake). Weitere Informationen finden Sie unter Verwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

- `frame` – Der zu schreibende `DataFrame`.
- `Database` – Die Data-Catalog-Datenbank, die die Tabelle enthält.
- `table_name` – Der Name der Data-Catalog-Tabelle, die dem Ziel zugeordnet ist.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Der zu verwendende Transformationskontext (optional).
- `additional_options` – Eine Sammlung optionaler Name/Wert-Paare.
  - `useSparkDataSink`— Wenn auf true gesetzt, zwingt AWS Glue, die native Spark Data Sink API zu verwenden, um in die Tabelle zu schreiben. Wenn Sie diese Option aktivieren, können Sie nach `additional_options` Bedarf alle Spark-Datenquellenoptionen hinzufügen. AWS Glue gibt diese Optionen direkt an den Spark-Writer weiter.
- `catalog_id` – Die Katalog-ID (Konto-ID) des Data Catalog, auf den zugegriffen wird. Wenn Sie keinen Wert angeben, wird die Standard-Konto-ID des Anrufers verwendet.

Einschränkungen

Berücksichtigen Sie die folgenden Einschränkungen, wenn Sie die `useSparkDataSink`-Option verwenden:

- Die `enableUpdateCatalog`-Option wird nicht unterstützt, wenn Sie die `useSparkDataSink`-Option verwenden.

Beispiel: Schreiben in eine Hudi-Tabelle mit dem Spark-Datenquellen-Writer

```
hudi_options = {
    'useSparkDataSink': True,
```

```
    'hoodie.table.name': <table_name>,
    'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
    'hoodie.datasource.write.recordkey.field': 'product_id',
    'hoodie.datasource.write.table.name': <table_name>,
    'hoodie.datasource.write.operation': 'upsert',
    'hoodie.datasource.write.precombine.field': 'updated_at',
    'hoodie.datasource.write.hive_style_partitioning': 'true',
    'hoodie.upsert.shuffle.parallelism': 2,
    'hoodie.insert.shuffle.parallelism': 2,
    'hoodie.datasource.hive_sync.enable': 'true',
    'hoodie.datasource.hive_sync.database': <database_name>,
    'hoodie.datasource.hive_sync.table': <table_name>,
    'hoodie.datasource.hive_sync.use_jdbc': 'false',
    'hoodie.datasource.hive_sync.mode': 'hms'}

glueContext.write_data_frame.from_catalog(
    frame = <df_product_inserts>,
    database = <database_name>,
    table_name = <table_name>,
    additional_options = hudi_options
)
```

write_dynamic_frame_from_jdbc_conf

`write_dynamic_frame_from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Schreibt und gibt einen `DynamicFrame` mit den angegebenen JDBC-Verbindungsinformationen zurück.

- `frame` – Der zu schreibende `DynamicFrame`.
- `catalog_connection` – Eine zu verwendende Katalogverbindung.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).
- `catalog_id` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei None wird die Standard-Konto-ID des Aufrufers verwendet.

write_from_jdbc_conf

`write_from_jdbc_conf(frame_or_dfc, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Schreibt und gibt eine `DynamicFrame` oder `DynamicFrameCollection` mit den angegebenen JDBC-Verbindungsinformationen zurück.

- `frame_or_dfc` – Der `DynamicFrame` oder die `DynamicFrameCollection`, der bzw. die geschrieben werden soll.
- `catalog_connection` – Eine zu verwendende Katalogverbindung.
- `connection_options` – Verbindungsoptionen, beispielsweise Pfad und Datenbanktabelle (optional). Weitere Informationen finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark.
- `redshift_tmp_dir` – Ein zu verwendendes temporäres Amazon-Redshift-Verzeichnis (optional).
- `transformation_ctx` – Ein zu verwendender Transformationskontext (optional).
- `catalog_id` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei None wird die Standard-Konto-ID des Aufrufers verwendet.

AWS Glue PySpark transformiert Referenz

AWS Glue bietet die folgenden integrierten Transformationen, die Sie in PySpark ETL-Vorgängen verwenden können. Ihre Daten werden von Transformation zu Transformation in einer Datenstruktur namens a DynamicFrameweitergeleitet. Dabei handelt es sich um eine Erweiterung von Apache Spark SQL`DataFrame`. Der `DynamicFrame` enthält Ihre Daten und Sie verweisen auf das Schema, um Ihre Daten zu verarbeiten.

Die meisten dieser Transformationen existieren auch als Methoden der `DynamicFrame`-Klasse. Weitere Informationen finden Sie unter DynamicFrame Transformationen.

- GlueTransform Basisklasse
- ApplyMapping Klasse
- DropFields Klasse
- DropNullFields Klasse
- ErrorsAsDynamicFrame Klasse

- EvaluateDataQuality Klasse
- FillMissingValues Klasse
- Filterklasse
- FindIncrementalMatches Klasse
- FindMatches Klasse
- FlatMap Klasse
- Join-Klasse
- Map-Klasse
- MapToCollection Klasse
- mergeDynamicFrame
- Relationalize-Klasse
- RenameField Klasse
- ResolveChoice Klasse
- SelectFields Klasse
- SelectFromCollection Klasse
- Klasse simplify_DDB_JSON
- Spigot-Klasse
- SplitFields Klasse
- SplitRows Klasse
- Unbox-Klasse
- UnnestFrame Klasse

GlueTransform Basisklasse

Die Basisklasse, von der alle `awsglue.transforms`-Klassen erben.

Die Klassen definieren alle eine `__call__`-Methode. Sie überschreiben entweder die `GlueTransform`-Klassenmethoden, die in den folgenden Abschnitten aufgeführt werden, oder werden standardmäßig mit dem Klassennamen aufgerufen.

Methoden

- apply(cls, *args, **kwargs)

- name(cls)
- describeArgs(cls)
- describeReturn(cls)
- describeTransform(cls)
- describeErrors(cls)
- describe(cls)

apply(cls, *args, **kwargs)

Wendet die Transformation an, indem die Transformationsklasse aufgerufen und das Ergebnis zurückgegeben wird.

- `cls` – Das `self`-Klassenobjekt.

name(cls)

Gibt den Namen der abgeleiteten Transformationsklasse zurück.

- `cls` – Das `self`-Klassenobjekt.

describeArgs(cls)

- `cls` – Das `self`-Klassenobjekt.

Gibt eine Liste der Wörterbücher zurück, die jeweils einem benannten Argument entsprechen. Dabei wird folgendes Format verwendet:

```
[
  {
    "name": "(name of argument)",
    "type": "(type of argument)",
    "description": "(description of argument)",
    "optional": "(Boolean, True if the argument is optional)",
    "defaultValue": "(Default value string, or None)(String; the default value, or
 None)"
  },
...
```

```
]
```

Löst eine `NotImplementedError`-Ausnahme aus, wenn der Aufruf in einer abgeleiteten Transformation erfolgt, in die nicht implementiert wurde.

describeReturn(cls)

- `cls` – Das `self`-Klassenobjekt.

Gibt ein Wörterbuch mit Informationen über den Rückgabetyp in folgendem Format zurück:

```
{
  "type": "(return type)",
  "description": "(description of output)"
}
```

Löst eine `NotImplementedError`-Ausnahme aus, wenn der Aufruf in einer abgeleiteten Transformation erfolgt, in die nicht implementiert wurde.

describeTransform(cls)

Gibt eine Zeichenfolge zur Beschreibung der Transformation zurück.

- `cls` – Das `self`-Klassenobjekt.

Löst eine `NotImplementedError`-Ausnahme aus, wenn der Aufruf in einer abgeleiteten Transformation erfolgt, in die nicht implementiert wurde.

describeErrors(cls)

- `cls` – Das `self`-Klassenobjekt.

Gibt eine Liste der Wörterbücher zurück, die jeweils eine mögliche Ausnahme beschreiben, die von dieser Transformation ausgelöst wird. Dabei wird folgendes Format verwendet:

```
[
  {
    "type": "(type of error)",
    "description": "(description of error)"
  },
```

```
...
]
```

describe(cls)

- `cls` – Das `self`-Klassenobjekt.

Gibt ein Objekt im folgenden Format zurück:

```
{
  "transform" : {
    "name" : cls.name( ),
    "args" : cls.describeArgs( ),
    "returns" : cls.describeReturn( ),
    "raises" : cls.describeErrors( ),
    "location" : "internal"
  }
}
```

ApplyMapping Klasse

Wendet ein Mapping in einem `DynamicFrame` an.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.apply_mapping()`-Methode zu verwenden, um Mapping auf einen `DynamicFrame` anzuwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie apply_mapping, um Felder umzubenennen und Feldtypen zu ändern.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__(frame, mappings, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Wendet ein deklaratives Mapping auf einen bestimmten `DynamicFrame` an.

- `frame` – Der `DynamicFrame`, in dem das Mapping angewendet werden soll (erforderlich).
- `mappings` – Eine Liste von Mapping-Tupeln (erforderlich). Jede besteht aus (Quellspalte, Quelltyp, Zielspalte, Zieltyp).

  Wenn die Quellspalte einen Punkt "`.`„ im Namen hat, müssen Sie Backticks "`` ` ``„ darum herum platzieren. Um beispielsweise `this.old.name` (Zeichenfolge) auf `thisNewName` abzubilden, würden Sie das folgende Tupel verwenden:

  ```
  ("`this.old.name`", "string", "thisNewName", "string")
  ```

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die mit Fehlern in der Transformation im Zusammenhang steht (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Gibt nur die Felder des `DynamicFrame` zurück, die in den „Mapping“-Tupeln angegeben sind.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

DropFields Klasse

Legt Felder in einem `DynamicFrame` ab.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.drop_fields()`-Methode zum Löschen von Feldern aus einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie drop_fields, um Felder aus einem `DynamicFrame` zu entfernen.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Legt Knoten in einem `DynamicFrame` ab.

- `frame` – Der `DynamicFrame`, in den die Knoten abgelegt werden sollen (erforderlich).
- `paths` – Eine Liste der vollständigen Pfade zu den abzulegenden Knoten (erforderlich).

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Gibt einen neuen `DynamicFrame` ohne die angegebenen Felder zurück.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

DropNullFields Klasse

Verwirft alle Nullfelder in einem `DynamicFrame` des Typs `NullType`. Dies sind Felder mit fehlenden oder Nullwerten in jedem Datensatz im `DynamicFrame`-Datensatz.

Beispiel

Dieses Beispiel verwendet `DropNullFields`, um einen neuen `DynamicFrame` zu erstellen, in dem Felder vom Typ `NullType` fallen gelassen wurden. Um `DropNullFields` zu demonstrieren, fügen wir eine neue Spalte `empty_column` vom Typ Null zum bereits geladenen `persons`-Datensatz hinzu.

> **Note**
>
> Informationen zum Zugriff auf den Datensatz, der in diesem Beispiel verwendet wird, finden Sie unter Codebeispiel: Verknüpfen und Inbeziehungsetzen von Daten und folgen Sie den Anweisungen in Schritt 1: Crawlen der Daten im Amazon S3 Bucket.

```
# Example: Use DropNullFields to create a new DynamicFrame without NullType fields

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from pyspark.sql.functions import lit
from pyspark.sql.types import NullType
from awsglue.dynamicframe import DynamicFrame
from awsglue.transforms import DropNullFields

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Add new column "empty_column" with NullType
persons_with_nulls = persons.toDF().withColumn("empty_column",
 lit(None).cast(NullType()))
persons_with_nulls_dyf = DynamicFrame.fromDF(persons_with_nulls, glueContext,
 "persons_with_nulls")
print("Schema for the persons_with_nulls_dyf DynamicFrame:")
persons_with_nulls_dyf.printSchema()
```

```
# Remove the NullType field
persons_no_nulls = DropNullFields.apply(persons_with_nulls_dyf)
print("Schema for the persons_no_nulls DynamicFrame:")
persons_no_nulls.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_with_nulls_dyf DynamicFrame:
root
|-- family_name: string
|-- name: string
```

```
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
|-- empty_column: null

null_fields ['empty_column']
Schema for the persons_no_nulls DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
```

```
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__(frame, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Verwirft alle Nullfelder in einem `DynamicFrame` des Typs `NullType`. Dies sind Felder mit fehlenden oder Nullwerten in jedem Datensatz im `DynamicFrame`-Datensatz.

- `frame` – Der `DynamicFrame`, in dem Nullfelder verworfen werden sollen (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Gibt einen neuen `DynamicFrame` ohne Nullfelder zurück.

apply(cls, *args, **kwargs)

- `cls` – cls

name(cls)

- `cls` – cls

describeArgs(cls)

- `cls` – cls

describeReturn(cls)

- `cls` – cls

describeTransform(cls)

- `cls` – cls

describeErrors(cls)

- `cls` – cls

describe(cls)

- `cls` – cls

ErrorsAsDynamicFrame Klasse

Rückgabe eines `DynamicFrame`, der verschachtelte Datensätze für Fehler enthält, die während der Erstellung der Quelle `DynamicFrame` aufgetreten sind.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.errorsAsDynamicFrame()`-Methode zum Abrufen und Anzeigen von Fehlerdatensätzen zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie errorsAsDynamic Frame, um Fehlerdatensätze anzuzeigen.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__Aufruf__(Frame)

Gibt einen `DynamicFrame` zurück, der verschachtelte Fehlerdatensätze enthält, die im Zusammenhang mit dem Quell-`DynamicFrame` stehen.

- `frame` – Der Quell-`DynamicFrame` (erforderlich).

apply(cls, *args, **kwargs)

- `cls` – cls

name(cls)

- `cls` – cls

describeArgs(cls)

- `cls` – cls

describeReturn(cls)

- `cls` – cls

describeTransform(cls)

- `cls` – cls

describeErrors(cls)

- `cls` – cls

describe(cls)

- `cls` – cls

EvaluateDataQuality Klasse

Wertet einen Datenqualitätsregelsatz anhand eines `DynamicFrame` aus und gibt ein neues `DynamicFrame` mit den Ergebnissen der Bewertung zurück.

Beispiel

Der folgende Beispielcode zeigt, wie die Datenqualität für ein `DynamicFrame` ausgewertet und dann die Datenqualitätsergebnisse angezeigt werden.

```
from awsglue.transforms import *
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsgluedq.transforms import EvaluateDataQuality

#Create Glue context
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
```

```
# Define DynamicFrame
legislatorsAreas = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="areas_json")

# Create data quality ruleset
ruleset = """Rules = [ColumnExists "id", IsComplete "id"]"""

# Evaluate data quality
dqResults = EvaluateDataQuality.apply(
    frame=legislatorsAreas,
    ruleset=ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "legislatorsAreas",
        "enableDataQualityCloudWatchMetrics": True,
        "enableDataQualityResultsPublishing": True,
        "resultsS3Prefix": "amzn-s3-demo-bucket1",
    },
)


# Inspect data quality results
dqResults.printSchema()
dqResults.toDF().show()
```

Output

```
root
|-- Rule: string
|-- Outcome: string
|-- FailureReason: string
|-- EvaluatedMetrics: map
|    |-- keyType: string
|    |-- valueType: double


+----------------------+-------+-------------+--------------------------------------+
|Rule                  |Outcome|FailureReason|EvaluatedMetrics                      |
+----------------------+-------+-------------+--------------------------------------+
|ColumnExists "id"     |Passed |null         |{}                                    |
|IsComplete "id"       |Passed |null         |{Column.first_name.Completeness -> 1.0}|
+----------------------+-------+-------------+--------------------------------------+
```

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (Frame, Regelsatz, publishing_options = {})

- `frame` – Die `DynamicFrame`, deren Datenqualität Sie bewerten möchten.
- `ruleset` – Ein Regelsatz der Definitionssprache für Datenqualität (DQDL) im Zeichenfolgen-Format. Weitere Informationen zu DQDL finden Sie im Referenz zu Data Quality Definition Language (DQDL)-Benutzerhandbuch.
- `publishing_options` – Ein Wörterbuch, das die folgenden Optionen zum Veröffentlichen von Auswertungsergebnissen und -metriken festlegt:
  - `dataQualityEvaluationContext`— Eine Zeichenfolge, die den Namespace angibt, unter dem AWS Glue Amazon CloudWatch Metriken und die Datenqualitätsergebnisse veröffentlichen soll. Die aggregierten Metriken werden in der Glue Studio-Oberfläche angezeigt CloudWatch, während die vollständigen Ergebnisse in der Benutzeroberfläche von AWS Glue Studio angezeigt werden.
    - Erforderlich: Nein
    - Standardwert: `default_context`
  - `enableDataQualityCloudWatchMetrics`— Gibt an, ob die Ergebnisse der Datenqualitätsbewertung veröffentlicht werden sollen. CloudWatch Mit der `dataQualityEvaluationContext`-Option geben Sie einen Namespace für die Metriken an.
    - Erforderlich: Nein
    - Standardwert: false
  - `enableDataQualityResultsPublishing` – Gibt an, ob die Datenqualitätsergebnisse auf der Registerkarte Data Quality (Datenqualität) in der Benutzeroberfläche von AWS Glue Studio angezeigt werden sollen.

  - Erforderlich: Nein
  - Standardwert: wahr
- `resultsS3Prefix`— Gibt den Amazon S3 S3-Speicherort an, an den AWS Glue die Ergebnisse der Datenqualitätsbewertung schreiben kann.
  - Erforderlich: Nein
  - Standardwert: "" (eine leere Zeichenfolge)

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FillMissingValues Klasse

Die `FillMissingValues`-Klasse lokalisiert Nullwerte und leere Zeichenfolgen in einem angegebenen `DynamicFrame` und verwendet Machine-Learning-Methoden wie lineare Regression und zufällige Gesamtstruktur, um die fehlenden Werte vorherzusagen. Der ETL-Job verwendet die

Werte im Eingabe-Dataset, um das Machine-Learning-Modell zu trainieren, das dann vorhersagt, was die fehlenden Werte sein sollten.

> **Tip**
>
> Wenn Sie inkrementelle Datensätze verwenden, wird jeder inkrementelle Satz als Trainingsdaten für das Machine-Learning-Modell verwendet, sodass die Ergebnisse möglicherweise nicht so genau sind.

Import:

```
from awsglueml.transforms import FillMissingValues
```

Methoden

- Anwenden

apply(frame, missing_values_column, output_column ="", transformation_ctx ="", info ="", stageThreshold = 0, totalThreshold = 0)

Trägt die fehlenden Werte eines Dynamic Frames in eine angegebene Spalte ein und gibt einen neuen Frame mit Schätzungen in einer neuen Spalte zurück. Bei Zeilen ohne fehlende Werte wird der Wert der angegebenen Spalte in die neue Spalte dupliziert.

- `frame` – Der `DynamicFrame`, in dem fehlende Werte ausgefüllt werden sollen. Erforderlich
- `missing_values_column` – Die Spalte, die fehlende Werte enthält (`null`-Werte und leere Zeichenfolgen). Erforderlich
- `output_column` – Der Name der neuen Spalte, die geschätzte Werte für alle Zeilen enthält, deren Wert gefehlt hat. Optional; der Standardwert ist der Wert von `missing_values_column` mit Suffix `"_filled"`.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional, Standardwert ist Null).

- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional, Standardwert ist Null).

Gibt einen neuen `DynamicFrame` mit einer zusätzlichen Spalte zurück, die Schätzungen für Zeilen mit fehlenden Werten und dem aktuellen Wert für andere Zeilen enthält.

Filterklasse

Erstellt einen neuen `DynamicFrame`, der Datensätzen aus der Eingabe `DynamicFrame` enthält, die eine angegebene Prädikat-Funktion erfüllen.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.filter()`-Methode zum Filtern von Datensätzen aus einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie Filter, um eine gefilterte Auswahl von Feldern zu erhalten.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0))

Gibt einen neuen `DynamicFrame` zurück, der durch Auswahl von Datensätzen aus dem Eingabe-`DynamicFrame` erstellt wurde, der eine angegebene Prädikat-Funktion erfüllt.

- `frame` – Der Quell-`DynamicFrame`, auf den die angegebene Filterfunktion angewendet werden soll (erforderlich).
- `f` – Die Prädikatfunktion, die auf jeden `DynamicRecord` im `DynamicFrame` angewendet werden soll. Die Funktion muss einen `DynamicRecord` als Argument enthalten und "True" zurückgeben, wenn der `DynamicRecord` die Filteranforderungen erfüllt, oder andernfalls "False" (erforderlich).

Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.

- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die mit Fehlern in der Transformation im Zusammenhang steht (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FindIncrementalMatches Klasse

Identifiziert übereinstimmende Datensätze im vorhandenen und inkrementellen `DynamicFrame` und erstellt einen neuen `DynamicFrame` mit einer eindeutigen Kennung, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

Import:

```
from awsglueml.transforms import FindIncrementalMatches
```

Methoden

- Anwenden

anwenden (existingFrame, incrementalFrame, transformId, transformation_ctx = „“, info = „“, stageThreshold = 0, totalThreshold = 0, enforcedMatches = none, Scores = 0) computeMatchConfidence

Identifiziert übereinstimmende Datensätze im `DynamicFrame` und erstellt einen neuen `DynamicFrame` mit einer eindeutigen Kennung, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

- `existingFrame`— Die vorhandenen und FindIncrementalMatches vorab abgeglichen`DynamicFrame`, um die Transformation anzuwenden. Erforderlich
- `incrementalFrame`— Das Inkrementelle`DynamicFrame`, auf das die FindIncrementalMatches Transformation angewendet werden soll, um sie mit der abzugleichen. `existingFrame` Erforderlich
- `transformId`— Die eindeutige ID, die der FindIncrementalMatches Transformation zugeordnet ist, die auf Datensätze in der `DynamicFrames` angewendet werden soll. Erforderlich
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Status-/Zustandsinformationen. Optional.
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation. Optional.
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird. Optional. Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird. Optional. Der Standardwert ist „Null“.

- `enforcedMatches` – Der `DynamicFrame`, der verwendet wird, um Übereinstimmungen zu erzwingen. Optional. Die Standardeinstellung ist None (Kein).
- `computeMatchConfidenceScores` – Ein boolescher Wert, der angibt, ob für jede Gruppe übereinstimmender Datensätze ein Konfidenzwert berechnet werden soll. Optional. Der Standardwert lautet „false“.

Gibt einen neuen `DynamicFrame` mit einer eindeutigen ID für jede Gruppe übereinstimmender Datensätze zurück.

FindMatches Klasse

Identifiziert übereinstimmende Datensätze im `DynamicFrame` und erstellt einen neuen `DynamicFrame` mit einer eindeutigen Kennung, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

Import:

```
from awsglueml.transforms import FindMatches
```

Methoden

- Anwenden

anwenden (frame, transformId, transformation_ctx = „“, info = „“, stageThreshold = 0, totalThreshold = 0, enforcedMatches = none, Scores = 0) computeMatchConfidence

Identifiziert übereinstimmende Datensätze im `DynamicFrame` und erstellt einen neuen `DynamicFrame` mit einer eindeutigen Kennung, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

- `frame`— `DynamicFrame` Das, um die Transformation anzuwenden. FindMatches Erforderlich
- `transformId`— Die eindeutige ID, die der FindMatches Transformation zugeordnet ist, die auf Datensätze in der angewendet werden sollDynamicFrame. Erforderlich
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Status-/Zustandsinformationen. Optional.
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation. Optional.
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird. Optional. Der Standardwert ist „Null“.

- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird. Optional. Der Standardwert ist „Null“.
- `enforcedMatches` – Der `DynamicFrame`, der verwendet wird, um Übereinstimmungen zu erzwingen. Optional. Die Standardeinstellung ist None (Kein).
- `computeMatchConfidenceScores` – Ein boolescher Wert, der angibt, ob für jede Gruppe übereinstimmender Datensätze ein Konfidenzwert berechnet werden soll. Optional. Der Standardwert lautet „false“.

Gibt einen neuen `DynamicFrame` mit einer eindeutigen ID für jede Gruppe übereinstimmender Datensätze zurück.

FlatMap Klasse

Wendet eine Transformation auf jeden `DynamicFrame` in einer Sammlung an. Die Ergebnisse werden nicht zu einem einzelnen `DynamicFrame` zusammengefasst, sondern als Sammlung aufbewahrt.

Beispiele für FlatMap

Der folgende Beispielausschnitt zeigt, wie die `ResolveChoice`-Transformation bei Anwendung auf einen `FlatMap` auf eine Sammlung dynamischer Frames angewendet wird. Die zur Eingabe verwendeten Daten befinden sich im JSON unter dem Platzhalter Amazon-S3-Adresse `s3://bucket/path-for-data/sample.json` und enthalten die folgenden Daten.

Beispiel für JSON-Daten

```
[{
    "firstname": "Arnav",
    "lastname": "Desai",
    "address": {
        "street": "6 Anyroad Avenue",
        "city": "London",
        "state": "England",
        "country": "UK"
    },
    "phone": 17235550101,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Independent Research",
        "Government Department of Examples"
```

```
    ]
},
{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
},
{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}]
```

Example Auf ResolveChoice eine anwenden DynamicFrameCollection und Ausgabe anzeigen.

```
#Read DynamicFrame
datasource = glueContext.create_dynamic_frame_from_options("s3", connection_options =
 {"paths":["s3://bucket/path/to/file/mysamplejson.json"]}, format="json")
datasource.printSchema()
datasource.show()

## Split to create a DynamicFrameCollection
```

```
split_frame=datasource.split_fields(["firstname","lastname","address"],"personal_info","busines
split_frame.keys()
print("---")

## Use FlatMap to run ResolveChoice
kwargs = {"choice": "cast:string"}
flat = FlatMap.apply(split_frame, ResolveChoice, frame_name="frame",
 transformation_ctx='tcx', **kwargs)
flat.keys()

##Select one of the DynamicFrames
personal_info = flat.select("personal_info")
personal_info.printSchema()
personal_info.show()
print("---")

business_info = flat.select("business_info")
business_info.printSchema()
business_info.show()
```

> **Important**
>
> Beim Aufruf von `FlatMap.apply` muss der `frame_name`-Parameter `"frame"` sein. Derzeit wird kein anderer Wert akzeptiert.

Beispielausgabe

```
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string
|-- phone: long
|-- affiliations: array
|    |-- element: string
---
{
    "firstname": "Mary",
```

```
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
---
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string

{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
```

```
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    }
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    }
}
---
root
|-- phone: long
|-- affiliations: array
|    |-- element: string

{
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
```

Methoden

- __call__

- Anwenden
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__ (dfc, frame_name BaseTransform, transformation_ctx = „“, **base_kwargs)

Wendet eine Transformation auf jeden `DynamicFrame` in einer Sammlung an und gleicht die Ergebnisse an.

- `dfc` – Die `DynamicFrameCollection`, über die die Angleichung erfolgen soll (erforderlich).
- `BaseTransform` – Eine von `GlueTransform` abgeleitete Transformation, die auf jedes Element der Sammlung angewandt werden soll (erforderlich).
- `frame_name` – Der Name des Arguments, an das die Elemente der Sammlung weitergeleitet werden sollen (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `base_kwargs` – Argumente, die an die Basistransformation weitergeleitet werden sollen (erforderlich).

Gibt eine neue `DynamicFrameCollection` zurück, die durch Anwenden der Transformation auf jeden `DynamicFrame` in der `DynamicFrameCollection`-Quelle erstellt wurde.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Join-Klasse

Führt einen Equality Join für zwei `DynamicFrames` durch.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.join()`-Methode zum Verbinden von `DynamicFrames`. Code-Beispiele finden Sie unter Beispiel: Verwenden von join zum Kombinieren von `DynamicFrames`.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__Aufruf__(frame1, frame2, keys1, keys2, transformation_ctx = "")

Führt einen Equality Join für zwei `DynamicFrames` durch.

- `frame1` – Der erste `DynamicFrame` für einen Join (erforderlich).
- `frame2` – Der zweite `DynamicFrame` für einen Join (erforderlich).
- `keys1` – Die Schlüssel für einen Join für den ersten Frame (erforderlich).
- `keys2` – Die Schlüssel für einen Join für den zweiten Frame (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

Gibt einen neuen `DynamicFrame` zurück, der durch die Verbindung der beiden `DynamicFrames` entsteht.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Map-Klasse

Erstellt einen neuen `DynamicFrame` durch Anwenden einer Funktion auf alle Datensätze in der Eingabe `DynamicFrame`.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.map()`-Methode zu verwenden, um eine Funktion alle Datensätze in einem `DynamicFrame` anzuwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden von map zur Anwendung einer Funktion auf jeden Datensatz in einem `DynamicFrame`.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Gibt einen neuen `DynamicFrame` zurück, der sich durch Anwenden der angegebenen Funktion auf alle `DynamicRecords` im ursprünglichen `DynamicFrame` ergibt.

- `frame` – Der ursprüngliche `DynamicFrame`, auf den die Zuweisungsfunktion angewendet werden muss (erforderlich).
- `f` – Die Funktion, die auf alle `DynamicRecords` im `DynamicFrame` angewendet werden soll. Die Funktion muss einen`DynamicRecord` als Argument enthalten und gibt einen neuen `DynamicRecord` zurück, der durch die Zuweisung erstellt wird (erforderlich).

  Ein `DynamicRecord` stellt einen logischen Datensatz in einem `DynamicFrame` dar. Er ist mit einer Zeile in einem Apache Spark `DataFrame` vergleichbar, mit der Ausnahme, dass er selbstbeschreibend ist und für Daten verwendet werden kann, die keinem festen Schema entsprechen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).

- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Gibt einen neuen `DynamicFrame` zurück, der sich durch Anwenden der angegebenen Funktion auf alle `DynamicRecords` im ursprünglichen `DynamicFrame` ergibt.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

MapToCollection Klasse

Wendet eine Transformation auf jeden `DynamicFrame` in der angegebenen `DynamicFrameCollection` an.

Methoden

- __call__

- Anwenden
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__ (dfc, frame_name BaseTransform, transformation_ctx = „“, **base_kwargs)

Wendet eine Transformationsfunktion auf jeden `DynamicFrame` in der angegebenen `DynamicFrameCollection` an.

- `dfc` – Die `DynamicFrameCollection`, über die die Transformationsfunktion angewendet werden muss (erforderlich).
- `callable` – Eine aufrufbare Transformationsfunktion, die auf jedes Element der Sammlung angewendet werden soll (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

Gibt eine neue `DynamicFrameCollection` zurück, die durch Anwenden der Transformation auf jeden `DynamicFrame` in der `DynamicFrameCollection`-Quelle erstellt wurde.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Relationalize-Klasse

Vereinfacht ein verschachteltes Schema in einem `DynamicFrame` und klappt Array-Spalten aus dem angeglichenen Frame aus.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.relationalize()`-Methode für einen Relationalize-Vorgang von `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie Relationalize, um ein verschachteltes Schema in einem `DynamicFrame` zu vereinfachen.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, staging_path=None, name='roottable', options=None, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Relationalisiert einen `DynamicFrame` und erzeugt eine Liste mit Frames, die generiert werden, indem die Verschachtelung verschachtelter Spalten aufgelöst und Array-Spalten ausgeklappt werden.

Sie können eine pivotierte Array-Spalte mithilfe des Join-Schlüssels, der während der Auflösung der Verschachtelung generiert wird, mit der Stammtabelle verbinden.

- `frame` – Der `DynamicFrame`, der relationalisiert werden soll (erforderlich).
- `staging_path` – Der Pfad, unter dem die Methode Partitionen pivotierter Tabellen im CSV-Format speichern kann (optional). Pivotierte Tabellen werden von diesem Pfad gelesen.
- `name` – Der Name der Stammtabelle (optional).
- `options` – Ein Wörterbuch der optionalen Parameter. Derzeit nicht verwendet.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

RenameField Klasse

Benennt einen Knoten innerhalb eines `DynamicFrame` um.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.rename_field()`-Methode zum Umbenennen eines Feldes in einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie rename_field, um Felder in einem `DynamicFrame` umzubenennen.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, old_name, new_name, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Benennt einen Knoten innerhalb eines `DynamicFrame` um.

- `frame` – Der `DynamicFrame`, in dem ein Knoten umbenannt werden soll (erforderlich).
- `old_name` – Der vollständige Pfad zu dem Knoten, den Sie umbenennen möchten (erforderlich).

  Wenn der alte Name Punkte enthält, funktioniert das nicht, RenameField es sei denn, Sie setzen Backticks um ihn herum ()```. Um zum Beispiel durch zu `this.old.name` ersetzen`thisNewName`, würden Sie RenameField wie folgt aufrufen:

```
newDyF = RenameField(oldDyF, "`this.old.name`", "thisNewName")
```

- `new_name` – Der neue Name, einschließlich des vollständigen Pfads (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

ResolveChoice Klasse

Löst einen Auswahltyp innerhalb eines `DynamicFrame` auf.

Beispiel

Wir empfehlen, dass Sie die `DynamicFrame.resolveChoice()`-Methode verwenden, um Felder zu behandeln, die mehrere Typen in einem `DynamicFrame` enthalten. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie ResolveChoice, um eine Spalte zu behandeln, die mehrere Typen enthält.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, specs = None, choice = "", transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Stellt Informationen zum Auflösen mehrdeutiger Typen innerhalb eines `DynamicFrame` bereit. Gibt den resultierenden `DynamicFrame` zurück.

- `frame` – Der `DynamicFrame`, in dem der Auswahltyp aufgelöst werden soll (erforderlich).
- `specs` – Eine Liste der aufzulösenden spezifischen Mehrdeutigkeiten, jede in Form eines Tupels:`(path, action)`. Der `path`-Wert identifiziert ein spezielles mehrdeutiges Element und der `action`-Wert identifiziert die entsprechende Auflösung.

  Sie können nur einen der Parameter `spec` und `choice` verwenden. Wenn der `spec`-Parameter nicht `None` ist, dann muss der `choice`-Parameter eine leere Zeichenfolge sein. Wenn umgekehrt der `choice`-Parameter keine leere Zeichenfolge ist, dann muss der `spec`-Parameter `None` sein. Wenn keiner der Parameter angegeben wird, versucht AWS Glue, das Schema zu analysieren und es zu verwenden, um Mehrdeutigkeiten aufzulösen.

  Sie können eine der folgenden Auflösungsstrategien im `action`-Teil eines `specs`-Tupels angeben:

- `cast` – Ermöglicht Ihnen, einen Typ anzugeben, in den umgewandelt werden soll (z. B. `cast:int`).
- `make_cols` – Löst eine potenzielle Mehrdeutigkeit durch Vereinfachen der Daten auf. Wenn `columnA` beispielsweise `int` oder `string` sein könnte, besteht die Auflösung darin, zwei Spalten mit den Namen `columnA_int` und `columnA_string` im resultierenden `DynamicFrame` zu erzeugen.
- `make_struct` – Löst eine potenzielle Mehrdeutigkeit durch Verwendung einer Struktur, um die Daten darzustellen. Wenn beispielsweise Daten in einer Spalte `int` oder `string` sein könnten, wird durch Verwendung der `make_struct`-Aktion eine Spalte von Strukturen im resultierenden `DynamicFrame` erzeugt, die sowohl ein `int` als auch ein `string` enthalten.
- `project` – Löst eine potenzielle Mehrdeutigkeit auf, indem nur Werte eines bestimmten Typs in dem resultierenden `DynamicFrame` beibehalten werden. Wenn beispielsweise Daten in einer `ChoiceType`-Spalte ein `int` oder eine `string` sein könnten, werden bei der Angabe einer `project:string`-Aktion Werte aus dem resultierenden `DynamicFrame` ausgelassen, die nicht vom Typ `string` sind.

Wenn `path` ein Array identifiziert, platzieren Sie leere eckige Klammern hinter dem Namen des Arrays, um eine Mehrdeutigkeit zu vermeiden. Angenommen, Sie arbeiten mit Daten, die wie folgt strukturiert sind:

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
]
```

Sie können anstelle der Zeichenfolgenversion die numerische Version des Preises auswählen, indem Sie den `path` auf `"myList[].price"` und die `action` auf `"cast:double"` setzen.

- `choice` – Die standardmäßige Auflösungsaktion, wenn der `specs`-Parameter `None` ist. Wenn der `specs`-Parameter nicht `None` ist, dann darf dies nur auf eine leere Zeichenfolge festgelegt werden.

  Zusätzlich zu den soeben beschriebenen `specs`-Aktionen unterstützt dieses Argument noch die folgende Aktion:

  - `MATCH_CATALOG` – Versucht jeden `ChoiceType` in einen entsprechenden Typ in der angegebenen Data-Catalog-Tabelle umzuwandeln.
- `database`— Die AWS Glue Data Catalog-Datenbank, die mit der `MATCH_CATALOG` Auswahl verwendet werden soll (erforderlich für`MATCH_CATALOG`).

- `table_name`— Der Name der AWS Glue Data Catalog-Tabelle, die mit der `MATCH_CATALOG` Aktion verwendet werden soll (erforderlich für`MATCH_CATALOG`).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

SelectFields Klasse

Die `SelectFields`-Klasse erstellt einen neuen `DynamicFrame` aus einem bestehenden `DynamicFrame` und behält nur die Felder bei, die Sie angeben. `SelectFields` bietet ähnliche Funktionen wie ein SQL `SELECT`-Nachricht.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.select_fields()`-Methode zum Auswählen von Feldern aus einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden von select_fields zum Erstellen eines neuen `DynamicFrame` mit ausgewählten Feldern.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Describe

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Ruft Felder (Knoten) in einem `DynamicFrame` ab.

- `frame` – Der `DynamicFrame` zur Auswahl von Feldern (erforderlich).
- `paths` – Eine Liste der vollständigen Pfade zu den auszuwählenden Feldern (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge, die mit Fehlern in der Transformation im Zusammenhang steht (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

Gibt einen neuen `DynamicFrame` zurück, der nur die angegebenen Felder enthält.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

SelectFromCollection Klasse

Wählt einen `DynamicFrame` in einer `DynamicFrameCollection` aus.

Beispiel

In diesem Beispiel wird `SelectFromCollection` verwendet, um einen `DynamicFrame` aus einer `DynamicFrameCollection` auszuwählen.

Beispieldatensatz

Das Beispiel wählt zwei `DynamicFrames` von einem `DynamicFrameCollection` mit dem Namen `split_rows_collection` aus. Die folgende Liste enthält die Schlüssel für `split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Beispiel-Code

```
# Example: Use SelectFromCollection to select
# DynamicFrames from a DynamicFrameCollection

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Select frames and inspect entries
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
frame_high.toDF().show()
```

Output

```
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
|  1|    0|                    fax|             202-225-3307|
|  1|    1|                  phone|             202-225-5731|
|  2|    0|                    fax|             202-225-3307|
|  2|    1|                  phone|             202-225-5731|
|  3|    0|                    fax|             202-225-3307|
|  3|    1|                  phone|             202-225-5731|
|  4|    0|                    fax|             202-225-3307|
|  4|    1|                  phone|             202-225-5731|
|  5|    0|                    fax|             202-225-3307|
|  5|    1|                  phone|             202-225-5731|
|  6|    0|                    fax|             202-225-3307|
|  6|    1|                  phone|             202-225-5731|
|  7|    0|                    fax|             202-225-3307|
|  7|    1|                  phone|             202-225-5731|
|  8|    0|                    fax|             202-225-3307|
```

```
|  8|    1|                   phone|             202-225-5731|
|  9|    0|                     fax|             202-225-3307|
|  9|    1|                   phone|             202-225-5731|
| 10|    0|                     fax|             202-225-6328|
| 10|    1|                   phone|             202-225-4576|
+---+-----+------------------------+-------------------------+
only showing top 20 rows

+---+-----+------------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+------------------------+-------------------------+
| 11|    0|                     fax|             202-225-6328|
| 11|    1|                   phone|             202-225-4576|
| 11|    2|                 twitter|           RepTrentFranks|
| 12|    0|                     fax|             202-225-6328|
| 12|    1|                   phone|             202-225-4576|
| 12|    2|                 twitter|           RepTrentFranks|
| 13|    0|                     fax|             202-225-6328|
| 13|    1|                   phone|             202-225-4576|
| 13|    2|                 twitter|           RepTrentFranks|
| 14|    0|                     fax|             202-225-6328|
| 14|    1|                   phone|             202-225-4576|
| 14|    2|                 twitter|           RepTrentFranks|
| 15|    0|                     fax|             202-225-6328|
| 15|    1|                   phone|             202-225-4576|
| 15|    2|                 twitter|           RepTrentFranks|
| 16|    0|                     fax|             202-225-6328|
| 16|    1|                   phone|             202-225-4576|
| 16|    2|                 twitter|           RepTrentFranks|
| 17|    0|                     fax|             202-225-6328|
| 17|    1|                   phone|             202-225-4576|
+---+-----+------------------------+-------------------------+
only showing top 20 rows
```

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn

- describeTransform
- describeErrors
- Beschreiben

__call__(dfc, key, transformation_ctx = "")

Ruft einen `DynamicFrame` aus einer `DynamicFrameCollection` ab.

- `dfc` – Die `DynamicFrameCollection`, aus der der `DynamicFrame` ausgewählt werden soll (erforderlich).
- `key` – Der Schlüssel des `DynamicFrame`, der ausgewählt werden soll (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Klasse simplify_DDB_JSON

Vereinfacht verschachtelte Spalten in a`DynamicFrame`, die sich speziell in der DynamoDB-JSON-Struktur befinden, und gibt eine neue vereinfachte Spalte zurück. `DynamicFrame`

Beispiel

Wir empfehlen, dass Sie die `DynamicFrame.simplify_ddb_json()` Methode verwenden, um verschachtelte Spalten in a zu vereinfachen`DynamicFrame`, die sich speziell in der DynamoDB-JSON-Struktur befinden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie simplify_ddb_json, um eine DynamoDB-JSON-Vereinfachung aufzurufen.

Spigot-Klasse

Schreibt Beispieldatensätze an ein bestimmtes Ziel, damit Sie die von Ihrem AWS Glue Job.

Beispiel

Es wird empfohlen, diese `DynamicFrame.spigot()`-Methode zu verwenden, um eine Teilmenge von Datensätzen von einem `DynamicFrame` an ein bestimmtes Ziel zu schreiben. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie Spigot, um Beispielfelder von einem `DynamicFrame` an Amazon S3 zu schreiben.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, path, options, transformation_ctx = "")

Schreibt Beispieldatensätze in ein angegebenes Ziel während einer Transformation.

- `frame` – Der `DynamicFrame` für den Spigot-Vorgang (erforderlich).
- `path` – Der Pfad zum Ziel, an das geschrieben werden soll (erforderlich).
- `options` – JSON-Schlüssel-Wert-Paare, die Optionen angeben (optional). Die `"topk"`-Option gibt an, dass die ersten k-Datensätze geschrieben werden sollen. Die `"prob"`-Option gibt an, wie wahrscheinlich es ist (in Form einer Dezimalzahl), dass ein bestimmter Datensatz ausgewählt wird. Sie verwenden sie, um Datensätze auszuwählen, die geschrieben werden sollen.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply

name(cls)

Geerbt von `GlueTransform` Name

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors

describe(cls)

Geerbt von `GlueTransform` Beschreiben

SplitFields Klasse

Teilt ein `DynamicFrame` über angegebene Felder in zwei neue.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.split_fields()`-Methode zum Abspalten von Feldern in einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie split_fields, um ausgewählte Felder in einen separaten `DynamicFrame` aufzuteilen.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, paths, name1 = None, name2 = None, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Spaltet ein oder mehrere Felder in einem `DynamicFrame` in einen neuen `DynamicFrame` ab und legt einen weiteren neuen `DynamicFrame` an, der die verbleibenden Felder enthält.

- `frame` – Die Quelle `DynamicFrame`, die in zwei neue Quellen aufgeteilt werden soll (erforderlich).
- `paths` – Eine Liste der vollständigen Pfade zu den zu teilenden Feldern (erforderlich).
- `name1` – Der Name, der dem `DynamicFrame` zugewiesen werden soll, der die zu trennenden Felder enthält (optional). Wird kein Name angegeben, wird der Name des Quell-Frames mit "1" angehängt.
- `name2` – Der Name, der dem `DynamicFrame` zugewiesen werden soll, der die Felder enthält, die nach dem Teilen der angegebenen Felder übrig bleiben (optional). Wird kein Name angegeben, wird der Name des Quell-Frames mit "2" angehängt.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.

- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

SplitRows Klasse

Erzeugt eine `DynamicFrameCollection`, die zwei `DynamicFrames` enthält. Ein `DynamicFrame` enthält nur die angegebenen Zeilen, die abgespalten werden sollen, und der zweite enthält alle restlichen Zeilen.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.split_rows()`-Methode zum Abspalten von Zeilen in einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie split_rows, um Zeilen in einem `DynamicFrame` abzuspalten.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, comparison_dict, name1="frame1", name2="frame2", transformation_ctx = "", info = None, stageThreshold = 0, totalThreshold = 0)

Teilt eine oder mehrere Zeilen in einem `DynamicFrame` auf einen neuen `DynamicFrame` auf.

- `frame` – Die Quelle `DynamicFrame`, die in zwei neue Quellen aufgeteilt werden soll (erforderlich).
- `comparison_dict` – Ein Wörterbuch, in dem der Schlüssel der vollständige Pfad zu einer Spalte ist und der Wert ein weiteres Wörterbuch für das Mapping von Vergleichsoperatoren zu Werten ist, mit denen der Spaltenwert verglichen wird. Beispielsweise teilt `{"age": {">": 10, "<": 20}}` Zeilen, bei denen der Wert "age" zwischen 10 und 20 liegt, exklusiv der Zeilen, bei denen "age" außerhalb dieses Bereichs liegt (erforderlich).
- `name1` – Der Name, der dem `DynamicFrame` zugewiesen werden soll, das die zu trennenden Zeilen enthält (optional).
- `name2` – Der Name, der dem `DynamicFrame` zugewiesen werden soll, das die Zeilen enthält, die nach dem Teilen der angegebenen Zeilen übrig bleiben (optional).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Unbox-Klasse

Entpackt (formatiert neu) ein Zeichenfolgenfeld in einem `DynamicFrame`.

Beispiel

Wir empfehlen Ihnen, die `DynamicFrame.unbox()`-Methode zum Entpacken eines Feldes in einem `DynamicFrame` zu verwenden. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie unbox, um ein Zeichenkettenfeld in eine Struktur zu entpacken.

Methoden

- __call__
- apply

- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, path, format, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0, **options)

Entpackt ein Zeichenfolgenfeld in ein `DynamicFrame`.

- `frame` – Der `DynamicFrame` in dem das Feld entpackt werden soll (erforderlich).
- `path` – Der vollständige Pfad zum `StringNode` zum Entpacken (erforderlich).
- `format` – Eine Formatspezifikation (optional). Dies wird für einen Amazon S3 verwendet oder AWS Glue Verbindung, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark.
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.
- `separator` – Ein Trenn-Token (optional).
- `escaper` – Ein Escape-Token (optional).
- `skipFirst` – `True` wenn die erste Datenzeile übersprungen werden soll, oder `False`, wenn sie nicht übersprungen werden soll (optional).
- withSchema – Eine Zeichenfolge mit dem Schema für die Daten, die entpackt werden sollen (optional). Diese sollte immer mit Hilfe von `StructType.json` erstellt werden.
- `withHeader` – `True` wenn die zu entpackenden Daten einen Header enthalten, oder `False`, wenn nicht (optional).

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

UnnestFrame Klasse

Löst einen `DynamicFrame` auf, vereinfacht verschachtelte Objekte in Elemente der obersten Ebene und generiert Join-Schlüssel für Array-Objekte.

Beispiel

Wir empfehlen, dass Sie die `DynamicFrame.unnest()`-Methode verwenden, um verschachtelte Strukturen in einem `DynamicFrame` zu vereinfachen. Code-Beispiele finden Sie unter Beispiel: Verwenden Sie unnest, um verschachtelte Felder in Felder der obersten Ebene umzuwandeln.

Methoden

- __call__

- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__(frame, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0)

Löst einen `DynamicFrame` auf, vereinfacht verschachtelte Objekte in Elemente der obersten Ebene und generiert Join-Schlüssel für Array-Objekte.

- `frame` – Der `DynamicFrame`, der aufgelöst werden soll (erforderlich).
- `transformation_ctx` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `info` – Eine Zeichenfolge im Zusammenhang mit Fehlern in der Transformation (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional). Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional). Der Standardwert ist „Null“.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FlagDuplicatesInColumn Klasse

Die `FlagDuplicatesInColumn` Transformation gibt eine neue Spalte mit einem bestimmten Wert in jeder Zeile zurück, der angibt, ob der Wert in der Quellspalte der Zeile mit einem Wert in einer früheren Zeile der Quellspalte übereinstimmt. Wenn Übereinstimmungen gefunden werden, werden sie als Duplikate gekennzeichnet. Das ursprüngliche Vorkommen wird nicht gekennzeichnet, da es nicht mit einer früheren Zeile übereinstimmt.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = column.FlagDuplicatesInColumn.apply(
        data_frame=datasource1,
        spark_context=sc,
        source_column="city",
        target_column="flag_col",
        true_string="True",
        false_string="False"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die `FlagDuplicatesInColumn` Transformation fügt eine neue Spalte `flag_col` zur `df_output` hinzu. DataFrame Diese Spalte wird einen Zeichenkettenwert enthalten, der angibt, ob die entsprechende Zeile einen doppelten Wert in der Spalte `city` hat oder nicht. Wenn eine Zeile einen doppelten `city`-Wert hat, enthält `flag_col` den `true_string`-Wert „True“. Wenn eine Zeile einen eindeutigen `city`-Wert hat, enthält `flag_col` den `false_string`-Wert „False“.

Das resultierende `df_output` DataFrame wird alle Spalten aus der ursprünglichen `datasource1` enthalten, plus die zusätzliche `flag_col`-Spalte, die doppelte `city`-Werte anzeigt. DataFrame

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_column, target_column, true_string=DEFAULT_TRUE_STRING, FALSE_STRING=default_false_string)

Die `FlagDuplicatesInColumn` Transformation gibt eine neue Spalte mit einem bestimmten Wert in jeder Zeile zurück, der angibt, ob der Wert in der Quellspalte der Zeile mit einem Wert in einer früheren Zeile der Quellspalte übereinstimmt. Wenn Übereinstimmungen gefunden werden, werden sie als Duplikate gekennzeichnet. Das ursprüngliche Vorkommen wird nicht gekennzeichnet, da es nicht mit einer früheren Zeile übereinstimmt.

- `source_column`— Name der Quellspalte.
- `target_column`— Name der Zielspalte.
- `true_string`— Zeichenfolge, die in die Zielspalte eingefügt werden soll, wenn ein Quellspaltenwert einen früheren Wert in dieser Spalte dupliziert.

- `false_string`— Zeichenfolge, die in die Zielspalte eingefügt werden soll, wenn sich ein Quellspaltenwert von früheren Werten in dieser Spalte unterscheidet.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FormatPhoneNumber Klasse

Die `FormatPhoneNumber` Transformation gibt eine Spalte zurück, in der eine Telefonnummernzeichenfolge in einen formatierten Wert umgewandelt wird.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *
```

```
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        ("408-341-5669",),
        ("4083415669",)
    ],
    ["phone"],
)

try:
    df_output = column_formatting.FormatPhoneNumber.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="phone",
        default_region="US"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird wie folgt aussehen:

```
```
+---------------+
| phone|
+---------------+
|(408) 341-5669|
|(408) 341-5669|
+---------------+
```
```

Die `FormatPhoneNumber` Transformation verwendet die `source_column` als `"phone"` und die `default_region` als `"US"`.

Bei der Transformation wurden beide Telefonnummern, unabhängig von ihrem ursprünglichen Format, erfolgreich in das US-Standardformat `(408) 341-5669` formatiert.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_column, phone_number_format=None, default_region=None, default_region_column=None)

Die Transformation `FormatPhoneNumber` gibt eine Spalte zurück, in der eine Telefonnummernzeichenfolge in einen formatierten Wert umgewandelt wird.

- `source_column` – Der Name einer vorhandenen Spalte.
- `phone_number_format`— Das Format, in das die Telefonnummer konvertiert werden soll. Wenn kein Format angegeben ist, ist `E.164` das Standardformat ein international anerkanntes Standardformat für Telefonnummern. Gültige Werte sind unter anderem:
  - E164 (lassen Sie den Punkt nach E weg)
- `default_region`— Eine gültige Regionalvorwahl, die aus zwei oder drei Großbuchstaben besteht und die Region für die Telefonnummer angibt, wenn in der Nummer selbst keine Landesvorwahl enthalten ist. Es `defaultRegionColumn` kann höchstens einer von `defaultRegion` oder angegeben werden.
- `default_region_column`— Der Name einer Spalte des erweiterten Datentyps`Country`. Der Regionalcode aus der angegebenen Spalte wird verwendet, um die Landesvorwahl für die Telefonnummer zu ermitteln, wenn in der Nummer selbst keine Landesvorwahl vorhanden ist. Es `defaultRegionColumn` kann höchstens einer von `defaultRegion` oder angegeben werden.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FormatCase Klasse

Die `FormatCase` Transformation ändert jede Zeichenfolge in einer Spalte in den angegebenen Groß- und Kleinschreibung.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = data_cleaning.FormatCase.apply(
        data_frame=datasource1,
```

```
        spark_context=sc,
        source_column="city",
        case_type="LOWER"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die `FormatCase` Transformation konvertiert die Werte in der Spalte `city` auf der Grundlage des Parameters `case_type="lower"` in Kleinbuchstaben. Das resultierende `df_output` DataFrame wird alle Spalten aus der ursprünglichen `datasource1` enthalten, jedoch mit den Werten der Spalte `city` in Kleinbuchstaben. DataFrame

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_column, case_type)

Die `FormatCase` Transformation ändert jede Zeichenfolge in einer Spalte in den angegebenen Falltyp.

- `source_column` – Der Name einer vorhandenen Spalte.
- `case_type`— Unterstützte Falltypen sind`CAPITAL`,`LOWER`,`UPPER`,`SENTENCE`.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FillWithMode Klasse

Die `FillWithMode` Transformation formatiert eine Spalte entsprechend dem von Ihnen angegebenen Telefonnummernformat. Sie können auch eine Tie-Breaker-Logik angeben, bei der einige Werte identisch sind. Betrachten Sie beispielsweise die folgenden Werte: `1 2 2 3 3 4`

Ein ModeType von `MINIMUM` bewirkt`FillWithMode`, dass 2 als Moduswert zurückgegeben wird. Wenn modeType den Wert 3 hat`MAXIMUM`, ist der Modus 3. Für `AVERAGE` ist der Modus 2,5.

Beispiel

```
from awsglue.context import *
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)
```

```
input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (1055.123, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.FillWithMode.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1",
        mode_type="MAXIMUM"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe des angegebenen Codes wird sein:

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
1055.123	13.12
1055.123	13.12
13.12	13.12
1055.123	13.12
+---------------+---------------+
```
```

Die `FillWithMode` Transformation aus dem Modul `awsglue.data_quality` wird auf das `input_df` angewendet. DataFrame Es ersetzt die `Null`-Werte in der Spalte durch den Maximalwert (`mode_type="MAXIMUM"`) aus den Nicht-Null-Werten in dieser `source_column_1` Spalte.

In diesem Fall ist der Maximalwert in der Spalte `1055,123`. `source_column_1` Daher werden die `Null`-Werte in in der Ausgabe `df_output` durch `source_column_1` `1055.123` ersetzt. DataFrame

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_column, mode_type)

Die Transformation formatiert die Groß- und Kleinschreibung von Zeichenketten in einer Spalte. `FillWithMode`

- `source_column` – Der Name einer vorhandenen Spalte.
- `mode_type`— Wie löst man Gleichheitswerte in den Daten auf? Dieser Wert muss einer von`MINIMUM`, `NONEAVERAGE`, oder sein`MAXIMUM`.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

FlagDuplicateRows Klasse

Die `FlagDuplicateRows` Transformation gibt eine neue Spalte mit einem bestimmten Wert in jeder Zeile zurück, der angibt, ob diese Zeile exakt mit einer früheren Zeile in der Datenmenge übereinstimmt. Wenn Übereinstimmungen gefunden werden, werden sie als Duplikate gekennzeichnet. Das ursprüngliche Vorkommen wird nicht gekennzeichnet, da es nicht mit einer früheren Zeile übereinstimmt.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.FlagDuplicateRows.apply(
        data_frame=input_df,
```

```
        spark_context=sc,
        target_column="flag_row",
        true_string="True",
        false_string="False",
        target_index=1
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird eine PySpark DataFrame mit einer zusätzlichen Spalte sein, die anhand der Spalte `flag_row` angibt, ob es sich bei einer Zeile um ein Duplikat handelt oder nicht. `source_column_1` Das resultierende `df_output` DataFrame wird die folgenden Zeilen enthalten:

```
```
+---------------+---------------+--------+
|source_column_1|source_column_2|flag_row|
+---------------+---------------+--------+
105.111	13.12	False
13.12	13.12	True
null	13.12	True
13.12	13.12	True
null	13.12	True
+---------------+---------------+--------+
```
```

Die `flag_row` Spalte gibt an, ob es sich bei einer Zeile um ein Duplikat handelt oder nicht. Die `true_string` ist auf „True“ gesetzt und die `false_string` ist auf „False“ gesetzt. Der `target_index` ist auf 1 gesetzt, was bedeutet, dass die `flag_row` Spalte an der zweiten Position (Index 1) in der Ausgabe eingefügt wird. DataFrame

Methoden

- __call__
- apply
- Name
- describeArgs

- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, target_column, true_string=DEFAULT_TRUE_STRING, FALSE_STRING=DEFAULT_FALSE_STRING, TARGET_INDEX=NONE)

Die Transformation `FlagDuplicateRows` gibt eine neue Spalte mit einem bestimmten Wert in jeder Zeile zurück, der angibt, ob diese Zeile exakt mit einer früheren Zeile im Datensatz übereinstimmt. Wenn Übereinstimmungen gefunden werden, werden sie als Duplikate gekennzeichnet. Das ursprüngliche Vorkommen wird nicht gekennzeichnet, da es nicht mit einer früheren Zeile übereinstimmt.

- `true_string`— Wert, der eingefügt werden soll, wenn die Zeile mit einer früheren Zeile übereinstimmt.
- `false_string`— Wert, der eingefügt werden soll, wenn die Zeile eindeutig ist.
- `target_column`— Name der neuen Spalte, die in den Datensatz eingefügt wird.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

RemoveDuplicates Klasse

Die `RemoveDuplicates` Transformation löscht eine ganze Zeile, wenn in einer ausgewählten Quellspalte ein doppelter Wert gefunden wird.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.RemoveDuplicates.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird a sein, PySpark DataFrame wobei Duplikate basierend auf der `source_column_1` Spalte entfernt werden. Das resultierende `df_output` DataFrame wird die folgenden Zeilen enthalten:

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
13.12	13.12
null	13.12
+---------------+---------------+
```
```

Beachten Sie, dass die Zeilen mit den `source_column_1` Werten `13.12` und `null` in der Ausgabe nur einmal vorkommen DataFrame, da die Duplikate anhand der Spalte entfernt wurden. `source_column_1`

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_column)

Die `RemoveDuplicates` Transformation löscht eine ganze Zeile, wenn in einer ausgewählten Quellspalte ein doppelter Wert gefunden wird.

- `source_column` – Der Name einer vorhandenen Spalte.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

MonthName Klasse

Die `MonthName` Transformation erstellt aus einer Zeichenfolge, die ein Datum darstellt, eine neue Spalte, die den Namen des Monats enthält.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

spark.conf.set("spark.sql.legacy.timeParserPolicy", "LEGACY")

input_df = spark.createDataFrame(
```

```
    [
        ("20-2018-12",),
        ("2018-20-12",),
        ("20182012",),
        ("12202018",),
        ("20122018",),
        ("20-12-2018",),
        ("12/20/2018",),
        ("02/02/02",),
        ("02 02 2009",),
        ("02/02/2009",),
        ("August/02/2009",),
        ("02/june/2009",),
        ("02/2020/june",),
        ("2013-02-21 06:35:45.658505",),
        ("August 02 2009",),
        ("2013/02/21",),
        (None,),
    ],
    ["column_1"],
)

try:
    df_output = datetime_functions.MonthName.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="column_1",
        target_column="target_column"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird sein:

```
```
+------------+------------+
| column_1|target_column|
+------------+------------+
|20-2018-12 | December |
```
```

```
|2018-20-12 | null |
| 20182012| null |
| 12202018| null |
| 20122018| null |
|20-12-2018 | December |
|12/20/2018 | December |
| 02/02/02 | February |
|02 02 2009 | February |
|02/02/2009 | February |
|August/02/2009| August |
|02/june/2009| null |
|02/2020/june| null |
|2013-02-21 06:35:45.658505| February |
|August 02 2009| August |
| 2013/02/21| February |
| null | null |
+------------+------------+
```

Die `MonthName` Transformation verwendet `source_column` als `"column_1"` und `target_column` als `"target_column"`. Sie versucht, den Monatsnamen aus der Zeichenfolge zu extrahieren, die ein unbekanntes Format hat oder nicht analysiert werden kann. Der Wert `"target_column"` ist auf `null` gesetzt. date/time strings in the `"column_1"` column and places it in the `"target_column"` column. If the date/time

Die Transformation extrahiert erfolgreich den Monatsnamen aus verschiedenen Datums-/Uhrzeitformaten, z. B. „20-12-2018", „20.12.2018", „02/02/2009", „2013-02-21 06:35:45.658 505" und „02. August 2009“.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

Die `MonthName` Transformation erstellt aus einer Zeichenfolge, die ein Datum darstellt, eine neue Spalte mit dem Namen des Monats.

- `source_column` – Der Name einer vorhandenen Spalte.
- `value`— Eine auszuwertende Zeichenfolge..
- `target_column`— Ein Name für die neu erstellte Spalte.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

IsEven Klasse

Die `IsEven` Transformation gibt einen booleschen Wert in einer neuen Spalte zurück, der angibt, ob die Quellspalte oder der Quellwert gerade ist. Wenn die Quellspalte oder der Quellwert eine Dezimalzahl ist, ist das Ergebnis falsch.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird sein:

```
```
+-------------+-------------+
|source_column|target_column|
+-------------+-------------+
5	Not even
0	Even
-1	Not even
2	Even
null	null
```
```

```
+------------+------------+
```
```

Die `IsEven` Transformation verwendet die `source_column` als „source_column“ und die `target_column` als „target_column“. Sie prüft, ob der Wert in der `"source_column"` gerade ist oder nicht. Wenn der Wert gerade ist, wird der Wert `"target_column"` auf den Wert `true_string` „Even“ gesetzt. Wenn der Wert ungerade ist, wird der Wert `"target_column"` auf `false_string` „Nicht gerade“ gesetzt. Falls der Wert `"source_column"` `Null` ist, wird der Wert `"target_column"` auf `null` gesetzt.

Die Transformation identifiziert die geraden Zahlen (0 und 2) korrekt und setzt den Wert `"target_column"` auf „Gerade“. Für ungerade Zahlen (5 und -1) wird der Wert `"target_column"` auf „Nicht gerade“ gesetzt. Für den Wert `Null` in `"source_column"` wird der Wert `"target_column"` auf `null` gesetzt.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, target_column, source_column=None, true_string=default_true_string, false_string=default_false_string, value=None)

Die Transformation `IsEven` gibt einen booleschen Wert in einer neuen Spalte zurück, der angibt, ob die Quellspalte oder der Quellwert gerade ist. Wenn die Quellspalte oder der Quellwert eine Dezimalzahl ist, ist das Ergebnis falsch.

- `source_column` – Der Name einer vorhandenen Spalte.
- `target_column`— Der Name der neuen Spalte, die erstellt werden soll.
- `true_string`— Eine Zeichenfolge, die angibt, ob der Wert gerade ist.
- `false_string`— Eine Zeichenfolge, die angibt, ob der Wert ungerade ist.
```

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

CryptographicHash Klasse

Die `CryptographicHash` Transformation wendet einen Algorithmus auf Hashwerte in der Spalte an.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

secret = "${SECRET}"
sc = SparkContext()
spark = SparkSession(sc)
```

```
input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_output = pii.CryptographicHash.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["id", "phone"],
        secret_id=secret,
        algorithm="HMAC_SHA256",
        output_format="BASE64",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird sein:

```
```
+---+------------+------------------+------------------+
| id| phone | id_hashed | phone_hashed |
+---+------------+------------------+------------------+
1	1234560000	QUI1zXTJiXmfIb...	juDBAmiRnnO3g...
2	1234560001	ZAUWiZ3dVTzCo...	vC8lgUqBVDMNQ...
3	1234560002	ZP4VvZWkqYifu...	Kl3QAkgswYpzB...
4	1234560003	3u8vO3wQ8EQfj...	CPBzK1P8PZZkV...
5	1234560004	eWkQJk4zAOIzx...	aLf7+mHcXqbLs...
```
```

```
| 6| 1234560005 | xtI9fZCJZCvsa... | dy2DFgdYWmr0p... |
| 7| 1234560006 | iW9hew7jnHuOf... | wwfGMCOEv6oOv... |
| 8| 1234560007 | H9V1pqvgkFhfS... | g9WKhagIXy9ht... |
| 9| 1234560008 | xDhEuHaxAUbU5... | b3uQLKPY+Q5vU... |
| 10| 1234560009 | GRN6nFXkxk349... | VJdsKt8VbxBbt... |
+---+------------+------------------+------------------+
```

Die Transformation berechnet die kryptografischen Hashes der Werte in den Spalten `id` und `phone` unter Verwendung des angegebenen Algorithmus und geheimen Schlüssels und codiert die Hashes im Base64-Format. Das resultierende `df_output` DataFrame enthält alle Spalten aus dem ursprünglichen `input_df` sowie die zusätzlichen Spalten `id_hashed` und `phone_hashed` mit den berechneten Hashes. DataFrame

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_columns, secret_id, algorithm=Keine, secret_version=Keine, create_secret_if_missing=False, output_format=Keine, entity_type_filter=Keine)

Die `CryptographicHash` Transformation wendet einen Algorithmus auf Hashwerte in der Spalte an.

- `source_columns`— Ein Array vorhandener Spalten.
- `secret_id`— Der ARN des geheimen Schlüssels von Secrets Manager. Der Schlüssel, der im Präfixalgorithmus für den Hash-basierten Nachrichtenauthentifizierungscode (HMAC) verwendet wird, um die Quellspalten zu hashen.
- `secret_version` Optional. Standardmäßig wird die neueste geheime Version verwendet.

- `entity_type_filter`— Optionales Array von Entitätstypen. Kann verwendet werden, um nur erkannte PII in einer Freitextspalte zu verschlüsseln.
- `create_secret_if_missing`— Optionaler boolescher Wert. Falls wahr, wird versucht, das Geheimnis im Namen des Aufrufers zu erstellen.
- `algorithm`— Der Algorithmus, der zum Hashing Ihrer Daten verwendet wird. Gültige Aufzählungswerte: MD5,, SHA1, HMAC_ SHA256 SHA512, HMAC_MD5, HMAC_, HMAC_SHA1. SHA256 SHA512

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Klasse entschlüsseln

Die `Decrypt` Transformation wird innerhalb von AWS Glue entschlüsselt. Ihre Daten können mit dem AWS Encryption SDK auch außerhalb von AWS Glue entschlüsselt werden. Wenn der angegebene

KMS-Schlüssel-ARN nicht mit dem übereinstimmt, der zum Verschlüsseln der Spalte verwendet wurde, schlägt der Entschlüsselungsvorgang fehl.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt = pii.Decrypt.apply(
        data_frame=df_encrypt,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt.show()
except:
```

```
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe erfolgt PySpark DataFrame mit der ursprünglichen Spalte `id` und der entschlüsselten Spalte `phone`:

```
```
+---+------------+
| id| phone|
+---+------------+
1	1234560000
2	1234560001
3	1234560002
4	1234560003
5	1234560004
6	1234560005
7	1234560006
8	1234560007
9	1234560008
10	1234560009
+---+------------+
```
```

Die `Encrypt` Transformation verwendet `source_columns` als `["phone"]` und `kms_key_arn` als Wert der Umgebungsvariablen `$ {KMS} `. Die Transformation verschlüsselt die Werte in der Spalte `phone` mit dem angegebenen KMS-Schlüssel. Das verschlüsselte DataFrame `df_encrypt` wird dann vom Modul `awsglue.pii` an die Transformation übergeben. `Decrypt` Es verwendet `source_columns` als `["phone"]` und `kms_key_arn` als Wert der Umgebungsvariablen `$ {KMS} `. Die Transformation entschlüsselt die verschlüsselten Werte in der Spalte `phone` mit demselben KMS-Schlüssel. Das resultierende `df_decrypt` DataFrame enthält die ursprüngliche Spalte `id` und die entschlüsselte Spalte `phone`.

Methoden

- __call__
- apply
- Name

- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_columns, kms_key_arn)

Die `Decrypt` Transformation wird innerhalb von AWS Glue entschlüsselt. Ihre Daten können mit dem AWS Encryption SDK auch außerhalb von AWS Glue entschlüsselt werden. Wenn der angegebene KMS-Schlüssel-ARN nicht mit dem übereinstimmt, der zum Verschlüsseln der Spalte verwendet wurde, schlägt der Entschlüsselungsvorgang fehl.

- `source_columns`— Ein Array vorhandener Spalten.
- `kms_key_arn`— Der Schlüssel-ARN des AWS Key Management Service-Schlüssels, der zum Entschlüsseln der Quellspalten verwendet werden soll.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Klasse verschlüsseln

Die `Encrypt` Transformation verschlüsselt Quellspalten mithilfe des AWS Key Management Service-Schlüssels. Die `Encrypt` Transformation kann bis zu 128 MiB pro Zelle verschlüsseln. Es wird versucht, das Format bei der Entschlüsselung beizubehalten. Um den Datentyp beizubehalten, müssen die Datentyp-Metadaten auf weniger als 1 KB serialisiert werden. Andernfalls müssen Sie den `preserve_data_type` Parameter auf False setzen. Die Metadaten des Datentyps werden im Verschlüsselungskontext im Klartext gespeichert.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
        spark_context=sc,
```

```
        source_columns=["phone"],
        kms_key_arn=kms
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe erfolgt in einer Spalte PySpark DataFrame mit der ursprünglichen `id`-Spalte und einer zusätzlichen Spalte, die die verschlüsselten Werte der `phone`-Spalte enthält.

```
```
+---+------------+------------------------+
| id| phone | phone_encrypted |
+---+------------+------------------------+
1	1234560000	EncryptedData1234...abc
2	1234560001	EncryptedData5678...def
3	1234560002	EncryptedData9012...ghi
4	1234560003	EncryptedData3456...jkl
5	1234560004	EncryptedData7890...mno
6	1234560005	EncryptedData1234...pqr
7	1234560006	EncryptedData5678...stu
8	1234560007	EncryptedData9012...vwx
9	1234560008	EncryptedData3456...yz0
10	1234560009	EncryptedData7890...123
+---+------------+------------------------+
```
```

Die `Encrypt` Transformation verwendet `source_columns` als `["phone"]` und `kms_key_arn` als Wert der Umgebungsvariablen `$ {KMS} `. Die Transformation verschlüsselt die Werte in der Spalte `phone` mit dem angegebenen KMS-Schlüssel. Das resultierende `df_encrypt` DataFrame enthält die ursprüngliche Spalte `id`, die ursprüngliche Spalte `phone` und eine zusätzliche Spalte namens `phone_encrypted`, die die verschlüsselten Werte der `phone`-Spalte enthält.

Methoden

- __call__
- apply
- Name

- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, source_columns, kms_key_arn, entity_type_filter=None, preserve_data_type=None)

Die `Encrypt` Transformation verschlüsselt Quellspalten mithilfe des AWS Key Management Service-Schlüssels.

- `source_columns`— Eine Reihe vorhandener Spalten.
- `kms_key_arn`— Der Schlüssel-ARN des AWS Key Management Service-Schlüssels, der zum Verschlüsseln der Quellspalten verwendet werden soll.
- `entity_type_filter`— Optionales Array von Entitätstypen. Kann verwendet werden, um nur erkannte PII in einer Freitextspalte zu verschlüsseln.
- `preserve_data_type`— Optionaler boolescher Wert. Standardwert ist „true“. Wenn der Wert falsch ist, wird der Datentyp nicht gespeichert.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

IntToIp Klasse

Die `IntToIp` Transformation konvertiert den Ganzzahlwert der Quellspalte oder einen anderen Wert in den entsprechenden IPv4 Wert in der Zielspalte und gibt das Ergebnis in einer neuen Spalte zurück.

Beispiel

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (3221225473,),
        (0,),
        (1,),
        (100,),
        (168430090,),
        (4294967295,),
        (4294967294,),
        (4294967296,),
        (-1,),
        (None,),
    ],
    ["source_column_int"],
)

try:
    df_output = web_functions.IntToIp.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_int",
```

```
        target_column="target_column",
        value=None
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Die Ausgabe wird sein:

```
```
+---------------+---------------+
|source_column_int|target_column|
+---------------+---------------+
3221225473	192.0.0.1
0	0.0.0.0
1	0.0.0.1
100	0.0.0.100
168430090	10.0.0.10
4294967295	255.255.255.255
4294967294	255.255.255.254
4294967296	null
-1	null
null	null
+---------------+---------------+
```
```

Die `IntToIp.apply` Transformation verwendet `source_column` als `"source_column_int"` und `target_column` als `"target_column"` und konvertiert die Ganzzahlwerte in der Spalte `source_column_int` in ihre entsprechende Adressdarstellung und speichert das Ergebnis in der Spalte `target_column`. IPv4

Gültige Ganzzahlwerte innerhalb des IPv4 Adressbereichs (0 bis 4294967295) werden von der Transformation erfolgreich in ihre IPv4 Adressdarstellung konvertiert (z. B. 192.0.0.1, 0.0.0.0, 10.0.0.10, 255.255.255.255).

Für Ganzzahlwerte außerhalb des gültigen Bereichs (z. B. 4294967296, -1) wird der Wert `target_column` auf `null` gesetzt. Für `Null`-Werte in der Spalte `source_column_int` wird der Wert `target_column` ebenfalls auf `null` gesetzt.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

Die `IntToIp` Transformation konvertiert den Integer-Wert der Quellspalte oder eines anderen Werts in den entsprechenden Wert in der Zielspalte und gibt das Ergebnis in einer neuen Spalte zurück. IPv4

- `sourceColumn` – Der Name einer vorhandenen Spalte.
- `value`— Eine auszuwertende Zeichenfolge.
- `targetColumn`— Der Name der neuen Spalte, die erstellt werden soll.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

IpToInt Klasse

Die `IpToInt` Transformation konvertiert den Wert der Internetprotokollversion 4 (IPv4) der Quellspalte oder einen anderen Wert in den entsprechenden Ganzzahlwert in der Zielspalte und gibt das Ergebnis in einer neuen Spalte zurück.

Beispiel

Für AWS Glue 4.0 und höher erstellen oder aktualisieren Sie Job-Argumente mit `key: --enable-glue-di-transforms, value: true`

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
    [
        ("192.0.0.1",),
        ("10.10.10.10",),
        ("1.2.3.4",),
        ("1.2.3.6",),
        ("http://12.13.14.15",),
        ("https://16.17.18.19",),
        ("1.2.3.4",),
        (None,),
        ("abc",),
        ("abc.abc.abc.abc",),
        ("321.123.123.123",),
        ("244.4.4.4",),
        ("255.255.255.255",),
```

```
    ],
    ["source_column_ip"],
)

    df_output = web_functions.IpToInt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_ip",
        target_column="target_column",
        value=None
    )
    df_output.show()
```

Output

Die Ausgabe wird sein:

```
```
+----------------+---------------+
|source_column_ip| target_column|
+----------------+---------------+
192.0.0.1	3221225473
10.10.10.10	168427722
1.2.3.4	16909060
1.2.3.6	16909062
http://12.13.14.15	null
https://16.17.18.19	null
1.2.3.4	16909060
null	null
abc	null
abc.abc.abc.abc	null
321.123.123.123	null
244.4.4.4	4102444804
255.255.255.255	4294967295
+----------------+---------------+
```
```

Die `IpToInt` Transformation verwendet `source_column` als `"source_column_ip"` und `target_column` als `"target_column"` und konvertiert die gültigen Adresszeichenketten in der Spalte `source_column_ip` in ihre entsprechende 32-Bit-Ganzzahldarstellung und speichert das Ergebnis in der Spalte `target_column`. IPv4

Für gültige IPv4 Adresszeichenfolgen (z. B. „192.0.0.1", „10.10.10", „1.2.3.4") konvertiert die Transformation sie erfolgreich in ihre Ganzzahldarstellung (z. B. 3221225473, 168427722, 16909060). Für Zeichenketten URLs, die keine gültigen IPv4 Adressen sind (z. B. Nicht-IP-Zeichenketten wie „abc“, ungültige IP-Formate wie „abc.abc.abc.abc“), wird der Wert `target_column` auf `null` gesetzt. Für `Null`-Werte in der Spalte `source_column_ip` wird der Wert `target_column` ebenfalls auf `null` gesetzt.

Methoden

- __call__
- apply
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Beschreiben

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

Die `IpToInt` Transformation konvertiert den Internet Protocol-Wert der Version 4 (IPv4) der Quellspalte oder einen anderen Wert in den entsprechenden Ganzzahlwert in der Zielspalte und gibt das Ergebnis in einer neuen Spalte zurück.

- `sourceColumn` – Der Name einer vorhandenen Spalte.
- `value`— Eine auszuwertende Zeichenfolge.
- `targetColumn`— Der Name der neuen Spalte, die erstellt werden soll.

apply(cls, *args, **kwargs)

Geerbt von `GlueTransform` apply.

name(cls)

Geerbt von `GlueTransform` Name.

describeArgs(cls)

Geerbt von `GlueTransform` describeArgs.

describeReturn(cls)

Geerbt von `GlueTransform` describeReturn.

describeTransform(cls)

Geerbt von `GlueTransform` describeTransform.

describeErrors(cls)

Geerbt von `GlueTransform` describeErrors.

describe(cls)

Geerbt von `GlueTransform` Beschreiben.

Datenintegration transformiert

Für AWS Glue 4.0 und höher erstellen oder aktualisieren Sie Job-Argumente mit`key: --enable-glue-di-transforms, value: true`.

Beispiel für ein Jobskript:

```
from pyspark.context import SparkContext

from awsgluedi.transforms import *
sc = SparkContext()

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
```

```
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Beispielsitzungen mit Notebooks

```
%idle_timeout 2880
%glue_version 4.0
%worker_type G.1X
%number_of_workers 5
%region eu-west-1
```

```
%%configure
{
    "--enable-glue-di-transforms": "true"
}
```

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
```

```
    print("Unexpected Error happened ")
    raise
```

Beispielsitzungen mit AWS CLI

```
aws glue create-session --default-arguments "--enable-glue-di-transforms=true"
```

DI-Transformationen:

- FlagDuplicatesInColumn Klasse
- FormatPhoneNumber Klasse
- FormatCase Klasse
- FillWithMode Klasse
- FlagDuplicateRows Klasse
- RemoveDuplicates Klasse
- MonthName Klasse
- IsEven Klasse
- CryptographicHash Klasse
- Klasse entschlüsseln
- Klasse verschlüsseln
- IntToIp Klasse
- IpToInt Klasse

Maven: Bündeln Sie das Plugin mit Ihren Spark-Anwendungen

Sie können die Transforms-Abhängigkeit mit Ihren Spark-Anwendungen und Spark-Distributionen (Version 3.3) bündeln, indem Sie die Plugin-Abhängigkeit in Ihrem Maven hinzufügen, `pom.xml` während Sie Ihre Spark-Anwendungen lokal entwickeln.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
    </repository>
</repositories>
```

```
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>AWSGlueTransforms</artifactId>
    <version>4.0.0</version>
</dependency>
```

Sie können die Binärdateien alternativ direkt von AWS Glue Maven-Artefakten herunterladen und sie wie folgt in Ihre Spark-Anwendung aufnehmen.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
AWSGlueTransforms/4.0.0/AWSGlueTransforms-4.0.0.jar -P /usr/lib/spark/jars/
```

Programmieren AWS Glue ETL-Skripte in Scala

Sie finden Scala-Codebeispiele und Hilfsprogramme für AWS Glue in der AWS Glue Proben-Repository auf der GitHub Website.

AWS Glue unterstützt eine Erweiterung des PySpark Scala-Dialekts für Scripting-Extraktions-, Transformations- und Ladeaufträge (ETL). In den folgenden Abschnitten wird beschrieben, wie Sie das verwenden AWS Glue Scala-Bibliothek und die AWS Glue API in ETL-Skripten und stellen Referenzdokumentation für die Bibliothek bereit.

Inhalt

- Mit Scala programmieren AWS Glue ETL-Skripte
  - Testen eines Scala-ETL-Programms in einem Jupyter Notebook auf einem Entwicklungsendpunkt
  - Testen eines Scala-ETL-Programms in einer Scala REPL
- Beispiel für Scala-Skripte – Streaming ETL
- APIs in der AWS Glue Scala-Bibliothek
  - com.amazonaws.services.glue
  - com.amazonaws.services.glue.ml
  - com.amazonaws.services.glue.dq
  - com.amazonaws.services.glue.types
  - com.amazonaws.services.glue.util
  - AWS Glue Scala ChoiceOption APIs

    - ChoiceOption Merkmal
    - ChoiceOption Objekt
        - Def apply
    - Fallklasse ChoiceOptionWithResolver
    - Fallklasse MatchCatalogSchemaChoiceOption
- Abstrakte DataSink Klasse
    - Def writeDynamicFrame
    - Def Rahmen pyWriteDynamic
    - Def writeDataFrame
    - Def Rahmen pyWriteData
    - Def setCatalogInfo
    - Def supportsFormat
    - Def setFormat
    - Def withFormat
    - Def setAccumulableSize
    - Def getOutputError RecordsAccumulable
    - Def Rahmen errorsAsDynamic
    - DataSink Objekt
        - def recordMetrics
- AWS Glue Scala-Merkmal DataSource
- AWS Glue Scala DynamicFrame APIs
    - AWS Glue Scala-Klasse DynamicFrame
        - Val errorsCount
        - Def applyMapping
        - Def assertErrorThreshold
        - Def count
        - Def dropField
        - Def dropFields
        - Def dropNulls
        - Def Rahmen errorsAsDynamic

- Def filter
- Def getName
- Def getNumPartitions
- Def Computer getSchemaIf
- Def isSchemaComputed
- Def javaToPython
- Def join
- Def map
- Def mergeDynamicFrames
- Def printSchema
- Def recomputeSchema
- Def relationalize
- Def renameField
- Def repartition
- Def resolveChoice
- Def schema
- Def selectField
- Def selectFields
- Def show
- Auf jeden Fall vereinfachen DDBJson
- Def spigot
- Def splitFields
- Def splitRows
- Def stageErrorsCount
- Def toDF
- Def unbox
- Def unnest
- Def unnest DDBJson
- Def withFrameSchema
- Def withName

        - Def withTransformationContext
    - Das DynamicFrame Objekt
        - Def apply
        - Def emptyDynamicFrame
        - Def fromPythonRDD
        - Def ignoreErrors
        - Def inlineErrors
        - Def-Fehler newFrameWith
- AWS Glue Scala-Klasse DynamicRecord
    - Def addField
    - Def dropField
    - Def setError
    - Def isError
    - Def getError
    - Def clearError
    - Def write
    - Def readFields
    - Def clone
    - Def schema
    - Def getRoot
    - Def toJson
    - Def getFieldNode
    - Def getField
    - Def hashCode
    - Def equals
    - DynamicRecord Objekt
        - Def apply
    - RecordTraverser Merkmal
- AWS Glue Scala GlueContext APIs
    - Spalten definieren addIngestionTime

    - def createDataFrame FromOptions
    - forEachBatch
    - def getCatalogSink
    - def getCatalogSource
    - def abrufen JDBCSink
    - def getSink
    - def-Format getSinkWith
    - def getSource
    - def-Format getSourceWith
    - def getSparkSession
    - def startTransaction
    - def commitTransaction
    - def cancelTransaction
    - def this
    - def this
    - def this
- MappingSpec
    - MappingSpec Fallklasse
    - MappingSpec Objekt
    - Val orderingByTarget
    - Def apply
    - Def apply
    - Def apply
- AWS Glue Scala ResolveSpec APIs
    - ResolveSpec Objekt
        - Def
        - Def
    - ResolveSpec Fallklasse
        - ResolveSpec Methoden definieren

- AWS Glue Scala ArrayNode APIs

    - ArrayNode Fallklasse
        - ArrayNode Methoden definieren
- AWS Glue Scala BinaryNode APIs
    - BinaryNode Fallklasse
        - BinaryNode Felder vom Typ Val
        - BinaryNode Methoden definieren
- AWS Glue Scala BooleanNode APIs
    - BooleanNode Fallklasse
        - BooleanNode Felder vom Typ Val
        - BooleanNode Methoden definieren
- AWS Glue Scala ByteNode APIs
    - ByteNode Fallklasse
        - ByteNode Felder vom Typ Val
        - ByteNode Methoden definieren
- AWS Glue Scala DateNode APIs
    - DateNode Fallklasse
        - DateNode Felder vom Typ Val
        - DateNode Methoden definieren
- AWS Glue Scala DecimalNode APIs
    - DecimalNode Fallklasse
        - DecimalNode Felder vom Typ Val
        - DecimalNode Methoden definieren
- AWS Glue Scala DoubleNode APIs
    - DoubleNode Fallklasse
        - DoubleNode Felder vom Typ Val
        - DoubleNode Methoden definieren
- AWS Glue Scala DynamicNode APIs
    - DynamicNode Klasse
    - DynamicNode Methoden definieren
    - DynamicNode Objekt

        - DynamicNode Methoden definieren
- EvaluateDataQuality Klasse
    - Def apply
    - Beispiel
- AWS Glue Scala FloatNode APIs
    - FloatNode Fallklasse
        - FloatNode Felder vom Typ Val
        - FloatNode Methoden definieren
- FillMissingValues Klasse
    - Def apply
- FindMatches Klasse
    - Def apply
- FindIncrementalMatches Klasse
    - Def apply
- AWS Glue Scala IntegerNode APIs
    - IntegerNode Fallklasse
        - IntegerNode Felder vom Typ Val
        - IntegerNode Methoden definieren
- AWS Glue Scala LongNode APIs
    - LongNode Fallklasse
        - LongNode Felder vom Typ Val
        - LongNode Methoden definieren
- AWS Glue Scala MapLikeNode APIs
    - MapLikeNode Klasse
        - MapLikeNode Methoden definieren
- AWS Glue Scala MapNode APIs
    - MapNode Fallklasse
        - MapNode Methoden definieren
- AWS Glue Scala NullNode APIs
    - NullNode Klasse

    - NullNode Case-Objekt
- AWS Glue Scala ObjectNode APIs
    - ObjectNode Objekt
        - ObjectNode Methoden definieren
    - ObjectNode Fallklasse
        - ObjectNode Methoden definieren
- AWS Glue Scala ScalarNode APIs
    - ScalarNode Klasse
        - ScalarNode Methoden definieren
    - ScalarNode Objekt
        - ScalarNode Methoden definieren
- AWS Glue Scala ShortNode APIs
    - ShortNode Fallklasse
        - ShortNode Felder vom Typ Val
        - ShortNode Methoden definieren
- AWS Glue Scala StringNode APIs
    - StringNode Fallklasse
        - StringNode Felder vom Typ Val
        - StringNode Methoden definieren
- AWS Glue Scala TimestampNode APIs
    - TimestampNode Fallklasse
        - TimestampNode Felder vom Typ Val
        - TimestampNode Methoden definieren
- AWS Glue Scala GlueArgParser APIs
    - GlueArgParser-Objekt
        - GlueArgParser Methoden definieren
- AWS Glue Job bei Scala APIs
    - Auftragsobjekt
        - Job def-Methoden

Mit Scala programmieren AWS Glue ETL-Skripte

Sie können automatisch ein Scala-Programm zum Extrahieren, Transformieren und Laden (ETL) generieren, indem Sie AWS Glue Konsole und ändern Sie es nach Bedarf, bevor Sie es einem Job zuweisen. Oder Sie können Ihr eigenes Programm von Grund auf neu schreiben. Weitere Informationen finden Sie unter Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue. AWS Glue kompiliert dann Ihr Scala-Programm auf dem Server, bevor der zugehörige Job ausgeführt wird.

Um sicherzustellen, dass Ihr Programm fehlerfrei kompiliert wird und wie erwartet ausgeführt wird, ist es wichtig, dass Sie es auf einen Entwicklungsendpunkt in einem REPL (Read-Eval-Print Loop) oder einem Jupyter Notebook laden und dort testen, bevor Sie es in einem Job ausführen. Da die Kompilierung auf einem Server stattfindet, haben Sie keinen guten Einblick in möglicherweise auftretende Probleme.

Testen eines Scala-ETL-Programms in einem Jupyter Notebook auf einem Entwicklungsendpunkt

Um ein Scala-Programm auf einem zu testen AWS Glue Entwicklungsendpunkt: Richten Sie den Entwicklungsendpunkt wie unter beschrieben ein. Hinzufügen eines Entwicklungsendpunkts

Als Nächstes verbinden Sie es mit einem Jupyter Notebook, das entweder lokal auf Ihrem Computer oder remote auf einem EC2 Amazon-Notebook-Server ausgeführt wird. Befolgen Sie zum Installieren einer lokalen Jupyter-Notebook-Version die Anweisungen unter Tutorial: Jupyter-Notizbuch in JupyterLab.

Der einzige Unterschied zwischen der Ausführung von Scala-Code und der Ausführung von PySpark Code auf Ihrem Notebook besteht darin, dass Sie jeden Absatz auf dem Notebook wie folgt beginnen sollten:

```
%spark
```

Dadurch wird verhindert, dass der Notebook-Server standardmäßig die PySpark Variante des Spark-Interpreters verwendet.

Testen eines Scala-ETL-Programms in einer Scala REPL

Sie können ein Scala-Programm auf einem Entwicklungsendpunkt testen, indem Sie AWS GlueScala REPL. Folgen Sie den Anweisungen unterTutorial: Verwenden Sie ein SageMaker KI-Notizbuch, außer am Ende des SSH-to-REPL Befehls, ersetzen Sie durch`-t gluepyspark`. `-t glue-spark-shell` Dies ruft den AWS Glue Scala REPL.

Geben Sie `sys.exit` ein, um die REPL zu schließen, wenn Sie fertig sind.

Beispiel für Scala-Skripte – Streaming ETL

Example

Das folgende Beispielskript stellt eine Verbindung mit Amazon Kinesis Data Streams her, verwendet ein Schema aus dem Data Catalog, um einen Datenstream zu analysieren, verknüpft den Stream mit einem statischen Dataset in Amazon S3 und gibt die verknüpften Ergebnisse im Parquet-Format in Amazon S3 aus.

```
// This script connects to an Amazon Kinesis stream, uses a schema from the data
 catalog to parse the stream,
// joins the stream to a static dataset on Amazon S3, and outputs the joined results to
 Amazon S3 in parquet format.
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import java.util.Calendar
import org.apache.spark.SparkContext
import org.apache.spark.sql.Dataset
import org.apache.spark.sql.Row
import org.apache.spark.sql.SaveMode
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.from_json
import org.apache.spark.sql.streaming.Trigger
import scala.collection.JavaConverters._

object streamJoiner {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val sparkSession: SparkSession = glueContext.getSparkSession
    import sparkSession.implicits._
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val staticData = sparkSession.read          // read() returns type DataFrameReader
      .format("csv")
      .option("header", "true")
      .load("s3://amzn-s3-demo-bucket/inputs/productsStatic.csv")  // load() returns a
 DataFrame
```

```
   val datasource0 = sparkSession.readStream   // readstream() returns type
DataStreamReader
     .format("kinesis")
     .option("streamName", "stream-join-demo")
     .option("endpointUrl", "https://kinesis.us-east-1.amazonaws.com")
     .option("startingPosition", "TRIM_HORIZON")
     .load                                     // load() returns a DataFrame

   val selectfields1 = datasource0.select(from_json($"data".cast("string"),
glueContext.getCatalogSchemaAsSparkSchema("stream-demos", "stream-join-demo2")) as
"data").select("data.*")

   val datasink2 = selectfields1.writeStream.foreachBatch { (dataFrame: Dataset[Row],
batchId: Long) => {   //foreachBatch() returns type DataStreamWriter
     val joined = dataFrame.join(staticData, "product_id")
     val year: Int = Calendar.getInstance().get(Calendar.YEAR)
     val month :Int = Calendar.getInstance().get(Calendar.MONTH) + 1
     val day: Int = Calendar.getInstance().get(Calendar.DATE)
     val hour: Int = Calendar.getInstance().get(Calendar.HOUR_OF_DAY)

     if (dataFrame.count() > 0) {
       joined.write                            // joined.write returns type
DataFrameWriter
         .mode(SaveMode.Append)
         .format("parquet")
         .option("quote", " ")
         .save("s3://amzn-s3-demo-bucket/output/" + "/year=" + "%04d".format(year)
+ "/month=" + "%02d".format(month) + "/day=" + "%02d".format(day) + "/hour=" +
"%02d".format(hour) + "/")
     }
   }
   }  // end foreachBatch()
     .trigger(Trigger.ProcessingTime("100 seconds"))
     .option("checkpointLocation", "s3://amzn-s3-demo-bucket/checkpoint/")
     .start().awaitTermination()             // start() returns type StreamingQuery
   Job.commit()
  }
}
```

APIs in der AWS Glue Scala-Bibliothek

AWS Glue unterstützt eine Erweiterung des PySpark Scala-Dialekts für Skript-Jobs zum Extrahieren, Transformieren und Laden (ETL). In den folgenden Abschnitten werden die in der APIs AWS Glue Scala-Bibliothek.

com.amazonaws.services.glue

Das com.amazonaws.services.glue-Paket im AWS Glue Die Scala-Bibliothek enthält Folgendes: APIs

- ChoiceOption
- DataSink
- DataSource Merkmal
- DynamicFrame
- DynamicRecord
- GlueContext
- MappingSpec
- ResolveSpec

com.amazonaws.services.glue.ml

Das com.amazonaws.services.glue.ml-Paket im AWS Glue Die Scala-Bibliothek enthält Folgendes: APIs

- FillMissingValues
- FindIncrementalMatches
- FindMatches

com.amazonaws.services.glue.dq

Das Paket com.amazonaws.services.glue.dq im AWS Glue Die Scala-Bibliothek enthält Folgendes: APIs

- EvaluateDataQuality

com.amazonaws.services.glue.types

Das com.amazonaws.services.glue.types-Paket im AWS Glue Die Scala-Bibliothek enthält Folgendes: APIs

- ArrayNode
- BinaryNode
- BooleanNode
- ByteNode
- DateNode
- DecimalNode
- DoubleNode
- DynamicNode
- FloatNode
- IntegerNode
- LongNode
- MapLikeNode
- MapNode
- NullNode
- ObjectNode
- ScalarNode
- ShortNode
- StringNode
- TimestampNode

com.amazonaws.services.glue.util

Das Paket com.amazonaws.services.glue.util im AWS Glue Die Scala-Bibliothek enthält Folgendes: APIs

- GlueArgParser
- Aufgabe

AWS Glue Scala ChoiceOption APIs

Themen

- ChoiceOption Merkmal
- ChoiceOption Objekt
- Fallklasse ChoiceOptionWithResolver
- Fallklasse MatchCatalogSchemaChoiceOption

Paket: com.amazonaws.services.glue

ChoiceOption Merkmal

```
trait ChoiceOption extends Serializable
```

ChoiceOption Objekt

ChoiceOption

```
object ChoiceOption
```

Eine allgemeine Strategie zur Lösung der Auswahl, die für alle `ChoiceType`-Knoten in einem `DynamicFrame` gilt.

- `val CAST`
- `val MAKE_COLS`
- `val MAKE_STRUCT`
- `val MATCH_CATALOG`
- `val PROJECT`

Def apply

```
def apply(choice: String): ChoiceOption
```

Fallklasse ChoiceOptionWithResolver

```
case class ChoiceOptionWithResolver(name: String, choiceResolver: ChoiceResolver)
 extends ChoiceOption {}
```

Fallklasse MatchCatalogSchemaChoiceOption

```
case class MatchCatalogSchemaChoiceOption() extends ChoiceOption {}
```

Abstrakte DataSink Klasse

Themen

- Def writeDynamicFrame
- Def Rahmen pyWriteDynamic
- Def writeDataFrame
- Def Rahmen pyWriteData
- Def setCatalogInfo
- Def supportsFormat
- Def setFormat
- Def withFormat
- Def setAccumulableSize
- Def getOutputError RecordsAccumulable
- Def Rahmen errorsAsDynamic
- DataSink Objekt

Paket: com.amazonaws.services.glue

```
abstract class DataSink
```

Der Schreiber ist analog zu einem `DataSource`. `DataSink` fasst ein Ziel und ein Format zusammen, in das ein `DynamicFrame` geschrieben werden kann.

Def writeDynamicFrame

```
def writeDynamicFrame( frame : DynamicFrame,
                       callSite : CallSite = CallSite("Not provided", "")
                     ) : DynamicFrame
```

Def Rahmen pyWriteDynamic

```
def pyWriteDynamicFrame( frame : DynamicFrame,
                         site : String = "Not provided",
                         info : String = "" )
```

Def writeDataFrame

```
def writeDataFrame(frame: DataFrame,
                   glueContext: GlueContext,
                   callSite: CallSite = CallSite("Not provided", "")
                   ): DataFrame
```

Def Rahmen pyWriteData

```
def pyWriteDataFrame(frame: DataFrame,
                     glueContext: GlueContext,
                     site: String = "Not provided",
                     info: String = ""
                     ): DataFrame
```

Def setCatalogInfo

```
def setCatalogInfo(catalogDatabase: String,
                   catalogTableName : String,
                   catalogId : String = "")
```

Def supportsFormat

```
def supportsFormat( format : String ) : Boolean
```

Def setFormat

```
def setFormat( format : String,
               options : JsonOptions
             ) : Unit
```

Def withFormat

```
def withFormat( format : String,
                options : JsonOptions = JsonOptions.empty
              ) : DataSink
```

Def setAccumulableSize

```
def setAccumulableSize( size : Int ) : Unit
```

Def getOutputError RecordsAccumulable

```
def getOutputErrorRecordsAccumulable : Accumulable[List[OutputError], OutputError]
```

Def Rahmen errorsAsDynamic

```
def errorsAsDynamicFrame : DynamicFrame
```

DataSink Objekt

```
object DataSink
```

def recordMetrics

```
def recordMetrics( frame : DynamicFrame,
                   ctxt : String
                 ) : DynamicFrame
```

AWS Glue Scala-Merkmal DataSource

Paket: com.amazonaws.services.glue

Eine High-Level-Schnittstelle zur Erzeugung eines `DynamicFrame`.

```
trait DataSource {

  def getDynamicFrame : DynamicFrame

  def getDynamicFrame( minPartitions : Int,
                       targetPartitions : Int
                     ) : DynamicFrame
  def getDataFrame : DataFrame

  /** @param num: the number of records for sampling.
    * @param options: optional parameters to control sampling behavior. Current
 available parameter for Amazon S3 sources in options:
    *  1. maxSamplePartitions: the maximum number of partitions the sampling will
 read.
    *  2. maxSampleFilesPerPartition: the maximum number of files the sampling will
 read in one partition.
    */
  def getSampleDynamicFrame(num:Int, options: JsonOptions = JsonOptions.empty):
 DynamicFrame

  def glueContext : GlueContext

  def setFormat( format : String,
                 options : String
               ) : Unit

  def setFormat( format : String,
                 options : JsonOptions
               ) : Unit

  def supportsFormat( format : String ) : Boolean

  def withFormat( format : String,
                  options : JsonOptions = JsonOptions.empty
                ) : DataSource
}
```

AWS Glue Scala DynamicFrame APIs

Paket: com.amazonaws.services.glue

Inhalt

- AWS Glue Scala-Klasse DynamicFrame
  - Val errorsCount
  - Def applyMapping
  - Def assertErrorThreshold
  - Def count
  - Def dropField
  - Def dropFields
  - Def dropNulls
  - Def Rahmen errorsAsDynamic
  - Def filter
  - Def getName
  - Def getNumPartitions
  - Def Computer getSchemaIf
  - Def isSchemaComputed
  - Def javaToPython
  - Def join
  - Def map
  - Def mergeDynamicFrames
  - Def printSchema
  - Def recomputeSchema
  - Def relationalize
  - Def renameField
  - Def repartition
  - Def resolveChoice
  - Def schema
  - Def selectField
  - Def selectFields

- Def show
- Auf jeden Fall vereinfachen DDBJson
- Def spigot
- Def splitFields
- Def splitRows
- Def stageErrorsCount
- Def toDF
- Def unbox
- Def unnest
- Def unnest DDBJson
- Def withFrameSchema
- Def withName
- Def withTransformationContext
- Das DynamicFrame Objekt
  - Def apply
  - Def emptyDynamicFrame
  - Def fromPythonRDD
  - Def ignoreErrors
  - Def inlineErrors
  - Def-Fehler newFrameWith

AWS Glue Scala-Klasse DynamicFrame

Paket: com.amazonaws.services.glue

```
class DynamicFrame extends Serializable with Logging  (
           val glueContext : GlueContext,
           _records : RDD[DynamicRecord],
           val name : String = s"",
           val transformationContext : String = DynamicFrame.UNDEFINED,
           callSite : CallSite = CallSite("Not provided", ""),
           stageThreshold : Long = 0,
           totalThreshold : Long = 0,
           prevErrors : => Long = 0,
```

```
            errorExpr : => Unit = {} )
```

Ein DynamicFrame ist eine verteilte Sammlung von selbstbeschreibenden DynamicRecord Objekten.

DynamicFrames wurden entwickelt, um ein flexibles Datenmodell für ETL-Operationen (Extrahieren, Transformieren und Laden) bereitzustellen. Sie benötigen kein Schema zum Erstellen, und Sie können damit Daten lesen und transformieren, die unstrukturierte oder inkonsistente Werte und Typen enthalten. Ein Schema kann bei Bedarf für solche Operationen berechnet werden, die eines benötigen.

DynamicFrames bieten eine Reihe von Transformationen für die Datenreinigung und ETL. Sie unterstützen auch die Konvertierung zu und von SparkSQL DataFrames zur Integration in vorhandenen Code und die vielen damit DataFrames verbundenen Analyseoperationen.

Die folgenden Parameter gelten für viele der AWS Glue Transformationen, die DynamicFrame s konstruieren:

- transformationContext – Der Bezeichner für diesen DynamicFrame. Der transformationContext wird als Schlüssel für den Auftrags-Lesezeichenstatus verwendet, der während der Ausführungen persistent ist.
- callSite – Liefert Kontextinformationen für die Fehlerberichterstattung. Diese Werte werden automatisch beim Aufruf von Python festgelegt.
- stageThreshold – Die maximale Anzahl der Fehlerdatensätze, die aufgrund der Berechnung dieses DynamicFrame zulässig sind, ehe eine Ausnahme ausgelöst wird. Ausgenommen sind Datensätze aus dem vorherigen DynamicFrame.
- totalThreshold – Die maximale Anzahl der Gesamtfehlersätze, bevor eine Ausnahme ausgelöst wird, einschließlich derjenigen aus früheren Frames.

Val errorsCount

```
val errorsCount
```

Die Anzahl der Fehlerdatensätze in diesem DynamicFrame. Dazu zählen Fehler aus früheren Operationen.

Def applyMapping

```
def applyMapping( mappings : Seq[Product4[String, String, String, String]],
                  caseSensitive : Boolean = true,
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
                  stageThreshold : Long = 0,
                  totalThreshold : Long = 0
                ) : DynamicFrame
```

- `mappings` – Eine Folge von Zuweisungen für die Erstellung eines neuen `DynamicFrame`.
- `caseSensitive` – Legt fest, ob bei Quellspalten die Groß-/Kleinschreibung beachtet wird. Dies auf false zu setzen, kann bei der Integration mit Speichern, bei denen Groß- und Kleinschreibung nicht beachtet wird, wie dem AWS Glue Data Catalog hilfreich sein.

Selektiert, projiziert und wandelt Spalten basierend auf Mappingreihenfolgen um.

Jede Zuweisung besteht aus einer Quellspalte und einem Typ und einer Zielspalte und einem Typ. Zuweisungen können entweder als „vierstelliger“ Tupel (`source_path`, `source_type`, `target_path`, `target_type`) oder als MappingSpec-Objekt, das dieselben Informationen enthält, angegeben werden.

Neben der Verwendung von Zuweisungen für einfache Projektionen und Umwandlungen können Sie diese auch zum Verschachteln oder Aufheben der Verschachtelung von Feldern verwenden, indem Sie Komponenten des Pfades mit „`.`“ (Punkt) trennen.

Angenommen, Sie haben einen `DynamicFrame` mit folgendem Schema:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- zip: int
}}}
```

Mit dem folgenden Aufruf können Sie die Verschachtelung der Felder `state` und `zip` aufheben:

```
{{{
```

```
  df.applyMapping(
   Seq(("name", "string", "name", "string"),
       ("age", "int", "age", "int"),
       ("address.state", "string", "state", "string"),
       ("address.zip", "int", "zip", "int")))
}}}
```

Das resultierende Schema lautet wie folgt:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- state: string
  |-- zip: int
}}}
```

Sie können auch `applyMapping` verwenden, um Spalten neu zu verschachteln. Durch folgende Operation wird beispielsweise die vorherige Transformation umgekehrt und eine neue Struktur namens `address` am Ziel erstellt:

```
{{{
  df.applyMapping(
   Seq(("name", "string", "name", "string"),
       ("age", "int", "age", "int"),
       ("state", "string", "address.state", "string"),
       ("zip", "int", "address.zip", "int")))
}}}
```

Feldnamen, die „`.`“ (Punkt-)Zeichen enthalten, können mit Hilfe von Backticks (`` ` ``) zitiert werden.

> **Note**
>
> Zurzeit können Sie die `applyMapping`-Methode nicht für die Zuweisung von Spalten verwenden, die unter Arrays verschachtelt sind.

Def assertErrorThreshold

```
def assertErrorThreshold : Unit
```

Eine Aktion, die eine Berechnung erzwingt und sicherstellt, dass die Anzahl der Fehlerdatensätze `stageThreshold` und `totalThreshold` nicht überschreitet. Löst eine Ausnahme aus, wenn eine der Bedingungen fehlschlägt.

Def count

```
lazy
def count
```

Gibt die Anzahl der Elemente in diesem `DynamicFrame` zurück.

Def dropField

```
def dropField( path : String,
               transformationContext : String = "",
               callSite : CallSite = CallSite("Not provided", ""),
               stageThreshold : Long = 0,
               totalThreshold : Long = 0
             ) : DynamicFrame
```

Gibt einen neuen `DynamicFrame` zurück, bei dem die angegebene Spalte entfernt wurde.

Def dropFields

```
def dropFields( fieldNames : Seq[String],   // The column names to drop.
                transformationContext : String = "",
                callSite : CallSite = CallSite("Not provided", ""),
                stageThreshold : Long = 0,
                totalThreshold : Long = 0
              ) : DynamicFrame
```

Gibt einen neuen `DynamicFrame` zurück, bei dem die angegebenen Spalten entfernt wurden.

Sie können diese Methode verwenden, um verschachtelte Spalten zu löschen, einschließlich derjenigen in Arrays. Sie kann aber nicht eingesetzt werden, um bestimmte Array-Elemente zu verwerfen.

Def dropNulls

```
def dropNulls( transformationContext : String = "",
               callSite : CallSite = CallSite("Not provided", ""),
               stageThreshold : Long = 0,
```

```
                totalThreshold : Long = 0 )
```

Gibt einen neuen DynamicFrame zurück, bei dem alle Nullspalten entfernt sind.

> **Note**
>
> Dies entfernt nur Spalten des Typs NullType. Einzelne Nullwerte in anderen Spalten werden weder entfernt noch geändert.

Def Rahmen errorsAsDynamic

```
def errorsAsDynamicFrame
```

Gibt einen neuen DynamicFrame mit den Fehlersätzen aus diesem DynamicFrame zurück.

Def filter

```
def filter( f : DynamicRecord => Boolean,
            errorMsg : String = "",
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Erstellt einen neuen DynamicFrame, der nur die Datensätze enthält, für die die Funktion „f“ true zurückgibt. Die Filterfunktion „f“ sollte den Eingabe-Datensatz nicht verändern.

Def getName

```
def getName : String
```

Gibt den Namen dieses DynamicFrame zurück.

Def getNumPartitions

```
def getNumPartitions
```

Gibt die Anzahl der Partitionen in diesem DynamicFrame zurück.

Def Computer getSchemaIf

```
def getSchemaIfComputed : Option[Schema]
```

Gibt das Schema zurück, wenn es bereits berechnet wurde. Scannt die Daten nicht, wenn das Schema noch nicht berechnet wurde.

Def isSchemaComputed

```
def isSchemaComputed : Boolean
```

Gibt "`true`" zurück, wenn das Schema für diesen `DynamicFrame` bereits berechnet wurde, andernfalls "`false`". Wenn diese Methode "false" zurückgibt, erfordert der Aufruf der `schema`-Methode eine erneute Übergabe dieser Datensätze im `DynamicFrame`.

Def javaToPython

```
def javaToPython : JavaRDD[Array[Byte]]
```

Def join

```
def join( keys1 : Seq[String],
          keys2 : Seq[String],
          frame2 : DynamicFrame,
          transformationContext : String = "",
          callSite : CallSite = CallSite("Not provided", ""),
          stageThreshold : Long = 0,
          totalThreshold : Long = 0
        ) : DynamicFrame
```

- `keys1` – Die Spalten in diesem `DynamicFrame`, die für den Join verwendet werden sollen.
- `keys2` – Die Spalten in `frame2`, die für den Join verwendet werden sollen. Muss die gleiche Länge haben wie `keys1`.
- `frame2` – Der andere `DynamicFrame` für einen Join.

Gibt das Ergebnis der Durchführung eines equijoin mit `frame2` über die angegebenen Schlüssel zurück.

Def map

```
def map( f : DynamicRecord => DynamicRecord,
         errorMsg : String = "",
         transformationContext : String = "",
         callSite : CallSite = CallSite("Not provided", ""),
         stageThreshold : Long = 0,
         totalThreshold : Long = 0
       ) : DynamicFrame
```

Gibt einen neuen `DynamicFrame` zurück, der erstellt wird, indem die angegebene Funktion „f“ auf jeden Datensatz in diesem `DynamicFrame` angewendet wird.

Diese Methode kopiert jeden Datensatz, ehe die angegebene Funktion angewendet wird, sodass das Verändern der Datensätze sicher ist. Wenn die Zuweisungsfunktion eine Ausnahme für einen bestimmten Datensatz auslöst, wird der Datensatz als fehlerhaft gekennzeichnet und der Stack-Trace als Spalte im Fehlerdatensatz gespeichert.

Def mergeDynamicFrames

```
def mergeDynamicFrames( stageDynamicFrame: DynamicFrame,  primaryKeys: Seq[String],
 transformationContext: String = "",
                         options: JsonOptions = JsonOptions.empty, callSite: CallSite =
 CallSite("Not provided"),
                         stageThreshold: Long = 0, totalThreshold: Long = 0):
 DynamicFrame
```

- `stageDynamicFrame` – Der Staging-`DynamicFrame`, der zusammengeführt werden soll.
- `primaryKeys` – Die Liste der Primärschlüsselfelder, mit denen Datensätze aus der Quelle und Staging-`DynamicFrames` übereinstimmen.
- `transformationContext` – Eine eindeutige Zeichenfolge, die zum Abrufen von Metadaten über die aktuelle Transformation verwendet wird (optional).
- `options` – Eine Zeichenfolge von JSON-Name-Wert-Paaren, die zusätzliche Informationen für diese Transformation bereitstellen.
- `callSite` – Wird verwendet, um Kontextinformationen für Fehlerberichte bereitzustellen.
- `stageThreshold` – Ein `Long`. Die Anzahl der Fehler in der angegebenen Transformation, für die die Verarbeitung fehlerhaft sein muss.

- `totalThreshold` – Ein `Long`. Die Gesamtzahl der Fehler bis einschließlich dieser Transformation, bei denen die Verarbeitung fehlerhaft sein muss.

Führt dieses `DynamicFrame` mit einem Staging-`DynamicFrame` basierend auf den angegebenen Primärschlüsseln zusammen, um Datensätze zu identifizieren. Doppelte Datensätze (Datensätze mit denselben Primärschlüsseln) werden nicht dedupliziert. Wenn kein übereinstimmender Datensatz im Staging-Frame vorhanden ist, werden alle Datensätze (einschließlich Duplikate) von der Quelle beibehalten. Wenn der Staging-Frame übereinstimmende Datensätze enthält, überschreiben die Datensätze aus dem Staging-Frame die Datensätze in der Quelle in AWS Glue.

Der zurückgegebene `DynamicFrame` enthält Datensatz A in folgenden Fällen:

1. Wenn sowohl im Quell- als auch im Staging-Frame `A` vorhanden ist, wird `A` im Staging-Frame zurückgegeben.
2. Wenn sich `A` in der Quelltabelle und `A.primaryKeys` nicht in `stagingDynamicFrame` befindet (d. h. `A` wird nicht in der Staging-Tabelle aktualisiert).

Der Quell- und der Staging-Frame müssen nicht dasselbe Schema haben.

Example

```
val mergedFrame: DynamicFrame = srcFrame.mergeDynamicFrames(stageFrame, Seq("id1",
 "id2"))
```

Def printSchema

```
def printSchema : Unit
```

Druckt das Schema dieses `DynamicFrame` in einem lesbaren Format in `stdout`.

Def recomputeSchema

```
def recomputeSchema : Schema
```

Erzwingt eine Neuberechnung des Schemas. Dies erfordert einen Scan über die Daten, aber es kann das Schema „verschärfen“, wenn es einige Felder im aktuellen Schema gibt, die nicht in den Daten vorhanden sind.

Gibt das neu berechnete Schema zurück.

Def relationalize

```
def relationalize( rootTableName : String,
                   stagingPath : String,
                   options : JsonOptions = JsonOptions.empty,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided"),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : Seq[DynamicFrame]
```

- `rootTableName` – Der Name für die Basis `DynamicFrame` in der Ausgabe. `DynamicFrames`, die durch das Zusammenfassen von Arrays erstellt werden, beginnen mit diesem als Präfix.
- `stagingPath` – Der Amazon-S3-Pfad (Amazon Simple Storage Service) zum Schreiben von Zwischendaten.
- `options` – Relationalisierung von Optionen und Konfiguration. Derzeit nicht verwendet.

Gleicht alle verschachtelten Strukturen an und pivotiert Arrays in separate Tabellen.

Mit dieser Operation können Sie tief verschachtelte Daten für die Aufnahme in eine relationale Datenbank vorbereiten. Verschachtelte Strukturen werden genauso wie die Unnest-Transformation auf eine Ebene gebracht. Außerdem werden Arrays in separate Tabellen pivotiert. Dabei wird jedes Array-Element zu einer Zeile. Angenommen, Sie haben einen `DynamicFrame` mit den folgenden Daten:

```
{"name": "Nancy", "age": 47, "friends": ["Fred", "Lakshmi"]}
{"name": "Stephanie", "age": 28, "friends": ["Yao", "Phil", "Alvin"]}
{"name": "Nathan", "age": 54, "friends": ["Nicolai", "Karen"]}
```

Führen Sie folgenden Code aus.

```
{{{
  df.relationalize("people", "s3:/my_bucket/my_path", JsonOptions.empty)
}}}
```

Es werden zwei Tabellen erstellt. Die erste Tabelle trägt den Namen „people“ und enthält Folgendes:

```
{{{
  {"name": "Nancy", "age": 47, "friends": 1}
  {"name": "Stephanie", "age": 28, "friends": 2}
  {"name": "Nathan", "age": 54, "friends": 3)
}}}
```

Das Freunde-Array wurde durch einen automatisch generierten Join-Schlüssel ersetzt. Eine separate Tabelle namens `people.friends` mit folgendem Inhalt wird erstellt:

```
{{{
  {"id": 1, "index": 0, "val": "Fred"}
  {"id": 1, "index": 1, "val": "Lakshmi"}
  {"id": 2, "index": 0, "val": "Yao"}
  {"id": 2, "index": 1, "val": "Phil"}
  {"id": 2, "index": 2, "val": "Alvin"}
  {"id": 3, "index": 0, "val": "Nicolai"}
  {"id": 3, "index": 1, "val": "Karen"}
}}}
```

In dieser Tabelle ist „`id`“ ein Join-Schlüssel, der identifiziert, aus welchem Datensatz das Array-Element stammt, „`index`“ bezieht sich auf die Position im ursprünglichen Array und „`val`“ steht für den tatsächlichen Array-Eintrag.

Die `relationalize`-Methode gibt die Sequenz von `DynamicFrames` zurück, die durch rekursives Anwenden dieses Prozesses auf alle Arrays erzeugt werden.

> **Note**
>
> Das Tool AWS Glue Die Bibliothek generiert automatisch Verbindungsschlüssel für neue Tabellen. Um sicherzustellen, dass Join-Schlüssel über alle Auftragsausführungen hinweg eindeutig sind, müssen Sie Auftrags-Lesezeichen aktivieren.

Def renameField

```
def renameField( oldName : String,
                 newName : String,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
```

```
                totalThreshold : Long = 0
              ) : DynamicFrame
```

- `oldName` – Der ursprüngliche Name der Spalte.
- `newName` – Der neue Name der Spalte.

Gibt einen neuen `DynamicFrame` zurück, wobei das angegebene Feld umbenannt ist.

Mit dieser Methode können Sie verschachtelte Felder umbenennen. Der folgende Code benennt beispielsweise innerhalb der Adressenstruktur `state` zu `state_code` um:

```
{{{
  df.renameField("address.state", "address.state_code")
}}}
```

Def repartition

```
def repartition( numPartitions : Int,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

Gibt einen neuen `DynamicFrame` mit `numPartitions` Partitionen zurück.

Def resolveChoice

```
def resolveChoice( specs : Seq[Product2[String, String]] = Seq.empty[ResolveSpec],
                   choiceOption : Option[ChoiceOption] = None,
                   database : Option[String] = None,
                   tableName : Option[String] = None,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided", ""),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : DynamicFrame
```

- `choiceOption` – Eine Aktion, die auf alle `ChoiceType`-Spalten anzuwenden ist, die nicht in der Spezifikationsreihenfolge aufgeführt sind.

- `database` – Die Data-Catalog-Datenbank zur Verwendung mit der `match_catalog`-Aktion.
- `tableName` – Die Data-Catalog-Tabelle zur Verwendung mit der `match_catalog`-Aktion.

Gibt einen neuen `DynamicFrame` zurück, indem eine oder mehrere `ChoiceTypes` durch einen spezifischeren Typ ersetzt werden.

Es gibt zwei Möglichkeiten für die Verwendung von `resolveChoice`. Die erste besteht in der Angabe bestimmter Spalten und der Art, wie diese aufgelöst werden. Sie werden als Tupels, bestehend aus (Spalte, Aktion)-Paaren, angegeben.

Im Folgenden sind die möglichen Aktionen aufgeführt:

- `cast:type` – Versucht, alle Werte in den angegebenen Typ umzuwandeln.
- `make_cols` – Konvertiert die einzelnen verschiedenen Typen in eine Spalte namens `columnName_type`.
- `make_struct` – Konvertiert eine Spalte in eine Struktur mit Schlüssel für die individuellen Typen.
- `project:type` – Behält nur Wert des angegebenen Typs bei.

Der andere Modus für `resolveChoice` dient zum Angeben einer einzigen Auflösung für alle `ChoiceTypes`. Sie können diesen verwenden, wenn die vollständige Liste der `ChoiceTypes` vor der Ausführung unbekannt ist. Zusätzlich zu den soeben aufgeführten Aktionen unterstützt dieser Modus noch die folgende Aktion:

- `match_catalogChoiceType` – Versucht jeden in einen entsprechenden Typ in der angegebenen Katalogtabelle umzuwandeln.

Beispiele:

Lösen Sie die `user.id`-Spalte auf, indem Sie eine Umwandlung in ein „int“ durchführen, und sorgen Sie dafür, dass das `address`-Feld nur Strukturen beibehält:

```
{{{
  df.resolveChoice(specs = Seq(("user.id", "cast:int"), ("address", "project:struct")))
}}}
```

Lösen Sie alle `ChoiceTypes` auf, indem Sie jede Auswahl in eine eigene Spalte umwandeln:

```
{{{
  df.resolveChoice(choiceOption = Some(ChoiceOption("make_cols")))
}}}
```

Lösen Sie alle `ChoiceTypes` auf, indem Sie diese in die Typen in der angegebenen Katalogtabelle umwandeln.

```
{{{
  df.resolveChoice(choiceOption = Some(ChoiceOption("match_catalog")),
                   database = Some("my_database"),
                   tableName = Some("my_table"))
}}}
```

Def schema

```
def schema : Schema
```

Gibt das Schema dieses `DynamicFrame` zurück.

Das zurückgegebene Schema enthält garantiert alle Felder in einem Datensatz in diesem `DynamicFrame`. In einigen wenigen Fällen kann es aber auch zusätzliche Felder enthalten. Die Unnest-Methode kann verwendet werden, um das Schema basierend auf den Datensätzen in diesem `DynamicFrame`zu „straffen“.

Def selectField

```
def selectField( fieldName : String,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

Gibt ein einzelnes Feld als `DynamicFrame` zurück.

Def selectFields

```
def selectFields( paths : Seq[String],
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
```

```
    stageThreshold : Long = 0,
    totalThreshold : Long = 0
  ) : DynamicFrame
```

- `paths` – Die Reihenfolge der zu wählenden Spaltennamen.

Gibt einen neuen `DynamicFrame` mit den angegebenen Spalten zurück.

> **Note**
>
> Sie können die `selectFields`-Methode nur verwenden, um Spalten der obersten Ebene auszuwählen. Sie können die applyMapping-Methode zum Auswählen verschachtelter Spalten einsetzen.

Def show

```
def show( numRows : Int = 20 ) : Unit
```

- `numRows` – Die Anzahl der zu druckenden Zeilen.

Druckt Zeilen aus diesem `DynamicFrame` im JSON-Format.

Auf jeden Fall vereinfachen DDBJson

DynamoDB-Exporte mit dem AWS Glue Der DynamoDB-Exportkonnektor führt zu JSON-Dateien mit bestimmten verschachtelten Strukturen. Weitere Informationen finden Sie unter Datenobjekte. `simplifyDDBJson` Vereinfacht verschachtelte Spalten in einem DynamicFrame solchen Datentyp und gibt eine neue vereinfachte DynamicFrame Spalte zurück. Wenn in einem Listentyp mehrere Typen oder ein Map-Typ enthalten ist, werden die Elemente in der Liste nicht vereinfacht. Diese Methode unterstützt nur Daten im DynamoDB-Export-JSON-Format. Erwägen `unnest` Sie, ähnliche Änderungen an anderen Datentypen vorzunehmen.

```
def simplifyDDBJson() : DynamicFrame
```

Diese Methode benötigt keine Parameter.

Beispieleingabe

Betrachten Sie das folgende Schema, das durch einen DynamoDB-Export generiert wurde:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

Beispiel-Code

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContextimport scala.collection.JavaConverters._

object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
```

```
    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "ddbTableARN",
        "dynamodb.s3.bucket" -> "exportBucketLocation",
        "dynamodb.s3.prefix" -> "exportBucketPrefix",
        "dynamodb.s3.bucketOwner" -> "exportBucketAccountID",
      ))
    ).getDynamicFrame()

    val simplified = dynamicFrame.simplifyDDBJson()
    simplified.printSchema()

    Job.commit()
  }

}
```

Beispielausgabe

Die Transformation `simplifyDDBJson` vereinfacht dies zu:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

Def spigot

```
def spigot( path : String,
            options : JsonOptions = new JsonOptions("{}"),
```

```
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Pass-Through-Transformation, die die gleichen Datensätze zurückgibt, darüber hinaus aber noch ein Subset an Datensätzen schreibt.

- `path` – Der Pfad in Amazon S3 zum Schreiben der Ausgabe in der Form `s3://bucket//path`.
- `options` – Eine optionale `JsonOptions`-Zuweisung, die das Sampling-Verhalten beschreibt.

Gibt einen `DynamicFrame` zurück, der die gleichen Datensätze wie dieser enthält.

Standardmäßig werden 100 willkürliche Datensätze in den durch `path` angegebenen Speicherort geschrieben. Sie können dieses Verhalten anpassen, indem Sie die `options`-Zuweisung verwenden. Gültige Schlüssel enthalten Folgendes.

- `topk` – Gibt die Gesamtzahl der geschriebenen Datensätze an. Der Standardwert ist 100.
- `prob` – Gibt an, wie wahrscheinlich es ist (in Form einer Dezimalzahl), dass ein einzelner Datensatz enthalten ist. Standard = 1.

Beispielsweise würde der folgende Aufruf den Datensatz sampeln, indem er jeden Datensatz mit einer Wahrscheinlichkeit von 20 % auswählt und nach dem Schreiben von 200 Datensätzen stoppt:

```
{{{
  df.spigot("s3://my_bucket/my_path", JsonOptions(Map("topk" -&gt; 200, "prob" -&gt;
 0.2)))
}}}
```

Def splitFields

```
def splitFields( paths : Seq[String],
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : Seq[DynamicFrame]
```

- `paths` – Die Pfade, die in den ersten `DynamicFrame` aufzunehmen sind.

Gibt eine Abfolge zweier `DynamicFrames` zurück. Die erste `DynamicFrame` enthält die angegebenen Pfade und die zweite alle anderen Spalten.

Beispiel

In diesem Beispiel wird eine aus der `persons` Tabelle in der `legislators` Datenbank im AWS Glue-Datenkatalog DynamicFrame erstellte Tabelle DynamicFrame in zwei Teile aufgeteilt, wobei die angegebenen Felder in das erste DynamicFrame und die verbleibenden Felder in ein zweites DynamicFrame gehen. Das Beispiel wählt dann das erste DynamicFrame aus dem Ergebnis aus.

```
val InputFrame = glueContext.getCatalogSource(database="legislators",
 tableName="persons",
transformationContext="InputFrame").getDynamicFrame()

val SplitField_collection = InputFrame.splitFields(paths=Seq("family_name", "name",
 "links.note",
"links.url", "gender", "image", "identifiers.scheme", "identifiers.identifier",
 "other_names.lang",
"other_names.note", "other_names.name"), transformationContext="SplitField_collection")

val ResultFrame = SplitField_collection(0)
```

Def splitRows

```
def splitRows( paths : Seq[String],
               values : Seq[Any],
               operators : Seq[String],
               transformationContext : String,
               callSite : CallSite,
               stageThreshold : Long,
               totalThreshold : Long
             ) : Seq[DynamicFrame]
```

Teilt Zeilen basierend auf Prädikaten, die Konstanten mit Spalten vergleichen, auf.

- `paths` – Die Spalten, die zum Vergleich verwendet werden sollen.
- `values` – Die konstanten Werte, die zum Vergleich verwendet werden sollen.
- `operators` – Die zum Vergleich zu verwendenden Operatoren.

Gibt eine Abfolge zweier DynamicFrames zurück. Die erste enthält Zeilen, für die das Prädikat "true" ist, und die zweite enthält solche, bei denen es "false" ist.

Prädikate werden über drei Sequenzen spezifiziert: „paths“ enthält die (evtl. geschachtelten) Spaltennamen, „values“ enthält die zu vergleichenden konstanten Werte und „operators“ enthält die Operatoren, die zum Vergleich verwendet werden sollen. Alle drei Sequenzen müssen gleich lang sein: Der n. Operator wird verwendet, um die n. Spalte mit dem n. Wert zu vergleichen.

Jeder Operator muss "!=", "=", "<=", "<", ">=" oder ">" sein.

Beispielsweise teilt der folgende Aufruf einen DynamicFrame so, dass der erste Ausgabe-Frame die Datensätze von Personen über 65 aus den USA enthält und der zweite alle anderen:

```
{{{
  df.splitRows(Seq("age", "address.country"), Seq(65, "USA"), Seq("&gt;=", "="))
}}}
```

Def stageErrorsCount

```
def stageErrorsCount
```

Gibt die Anzahl der Fehlerdatensätze zurück, die bei der Berechnung dieses DynamicFrame generiert wurden. Ausgenommen sind Fehler aus vorherigen Operationen, die diesem DynamicFrame als Eingabe übergeben wurden.

Def toDF

```
def toDF( specs : Seq[ResolveSpec] = Seq.empty[ResolveSpec] ) : DataFrame
```

Konvertiert DynamicFrame in einen Apache Spark SQL DataFrame mit demselben Schema und denselben Datensätzen.

> **Note**
>
> Da DataFrames ChoiceTypes nicht unterstützen, konvertiert diese Methode ChoiceType-Spalten automatisch in StructTypes. Weitere Informationen und Optionen zur Auflösung der Auswahl finden Sie unter resolveChoice.

Def unbox

```
def unbox( path : String,
           format : String,
           optionString : String = "{}",
           transformationContext : String = "",
           callSite : CallSite = CallSite("Not provided"),
           stageThreshold : Long = 0,
           totalThreshold : Long = 0
         ) : DynamicFrame
```

- `path` – Die zu analysierende Spalte. Muss eine Zeichenfolge oder ein Binärwert sein.
- `format` – Das für die Analyse zu verwendende Format.
- `optionString` – An das Format zu übergebende Optionen, beispielsweise das CSV-Trennzeichen.

Analysiert eine eingebettete Zeichenfolge oder eine binäre Spalte entsprechend des angegebenen Formats. Analysierte Spalten werden unterhalb einer Struktur mit dem ursprünglichen Spaltennamen verschachtelt.

Angenommen, Sie haben eine CSV-Datei mit einer eingebetteten JSON-Spalte.

```
name, age, address
Sally, 36, {"state": "NE", "city": "Omaha"}
...
```

Nach einer ersten Analyse erhalten Sie einen `DynamicFrame` mit folgendem Schema:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: string
}}}
```

Sie können `unbox` für die Adressspalte aufrufen, um die einzelnen Komponenten zu analysieren.

```
{{{
  df.unbox("address", "json")
```

```
}}}
```

Dadurch erhalten wir einen `DynamicFrame` mit folgendem Schema:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

Def unnest

```
def unnest( transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Gibt einen neuen `DynamicFrame` zurück, bei dem allen verschachtelten Strukturen auf eine Ebene gebracht wurden. Namen werden mit Hilfe des „`.`“ (Punkt-)Zeichens erstellt.

Angenommen, Sie haben einen `DynamicFrame` mit folgendem Schema:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

Der folgende Aufruf hebt die Einbettung der Adressenstruktur auf:

```
{{{
  df.unnest()
}}}
```

Das resultierende Schema lautet wie folgt:

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address.state: string
  |-- address.city: string
}}}
```

Diese Methode verhindert auch verschachtelte Strukturen innerhalb von Arrays. Aus historischen Gründen werden den Namen solcher Felder jedoch der Name des umschließenden Arrays und „`.val`“ vorangestellt.

Def unnest DDBJson

```
unnestDDBJson(transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0): DynamicFrame
```

Hebt die Verschachtelung der Spalten in einem `DynamicFrame` auf, die sich speziell in der DynamoDB-JSON-Struktur befinden, und gibt einen neuen, nicht verschachtelten `DynamicFrame` zurück. Bei Spalten, die aus einem Array von Strukturtypen bestehen, wird die Verschachtelung nicht aufgehoben. Dies ist ein spezieller Typ der Transformation zum Aufheben der Verschachtelung, der sich anders verhält als die reguläre `unnest`-Transformation und erfordert, dass sich die Daten bereits in der DynamoDB-JSON-Struktur befinden. Weitere Informationen finden Sie unter DYNAMODB JSON.

Das Schema eines Vorgangs zum Lesen eines Exports mit der DynamoDB-JSON-Struktur könnte beispielsweise wie folgt aussehen:

```
root
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
|    |    |-- N: string
|    |-- ColD: struct
```

```
|    |    |-- L: array
|    |    |    |-- element: null
```

Die unnestDDBJson()-Transformation würde dies folgendermaßen umwandeln:

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

Das folgende Codebeispiel zeigt, wie Sie den AWS Glue DynamoDB-Exportconnector verwenden, einen DynamoDB-JSON-Unnest aufrufen und die Anzahl der Partitionen ausgeben:

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "<test_source>",
        "dynamodb.s3.bucket" -> "<bucket name>",
        "dynamodb.s3.prefix" -> "<bucket prefix>",
        "dynamodb.s3.bucketOwner" -> "<account_id of bucket>",
      ))
    ).getDynamicFrame()

    val unnested = dynamicFrame.unnestDDBJson()
```

```
    print(unnested.getNumPartitions())

    Job.commit()
  }

}
```

Def withFrameSchema

```
def withFrameSchema( getSchema : () => Schema ) : DynamicFrame
```

- `getSchema` – Eine Funktion, die das zu verwendende Schema zurückgibt. Wird als Null-Parameter-Funktion angegeben, um eine möglicherweise kostenintensive Berechnung zu verhindern.

Legt das Schema dieses `DynamicFrame` auf den angegebenen Wert fest. Dies wird in erster Linie intern verwendet, um eine kostspielige Neuberechnung des Schemas zu vermeiden. Das übergebene Schema muss alle Spalten enthalten, die in den Daten vorhanden sind.

Def withName

```
def withName( name : String ) : DynamicFrame
```

- `name` – Der zu verwendende neue Name.

Gibt eine Kopie dieser `DynamicFrame` mit einem neuen Namen zurück.

Def withTransformationContext

```
def withTransformationContext( ctx : String ) : DynamicFrame
```

Gibt eine Kopie dieser `DynamicFrame` mit dem angegebenen Transformationskontext zurück.

Das DynamicFrame Objekt

Paket: com.amazonaws.services.glue

```
object DynamicFrame
```

Def apply

```
def apply( df : DataFrame,
           glueContext : GlueContext
         ) : DynamicFrame
```

Def emptyDynamicFrame

```
def emptyDynamicFrame( glueContext : GlueContext ) : DynamicFrame
```

Def fromPythonRDD

```
def fromPythonRDD( rdd : JavaRDD[Array[Byte]],
                   glueContext : GlueContext
                 ) : DynamicFrame
```

Def ignoreErrors

```
def ignoreErrors( fn : DynamicRecord => DynamicRecord ) : DynamicRecord
```

Def inlineErrors

```
def inlineErrors( msg : String,
                  callSite : CallSite
                ) : (DynamicRecord => DynamicRecord)
```

Def-Fehler newFrameWith

```
def newFrameWithErrors( prevFrame : DynamicFrame,
                        rdd : RDD[DynamicRecord],
                        name : String = "",
                        transformationContext : String = "",
                        callSite : CallSite,
                        stageThreshold : Long,
                        totalThreshold : Long
```

```
                        ) : DynamicFrame
```

AWS Glue Scala-Klasse DynamicRecord

Themen

- Def addField
- Def dropField
- Def setError
- Def isError
- Def getError
- Def clearError
- Def write
- Def readFields
- Def clone
- Def schema
- Def getRoot
- Def toJson
- Def getFieldNode
- Def getField
- Def hashCode
- Def equals
- DynamicRecord Objekt
- RecordTraverser Merkmal

Paket: com.amazonaws.services.glue

```
class DynamicRecord extends Serializable with Writable with Cloneable
```

Ein `DynamicRecord` ist eine selbstbeschreibende Datenstruktur, die eine Reihe von Daten im bearbeiteten Datenbestand repräsentiert. Sie ist selbstbeschreibend in dem Sinne, dass Sie das Schema der Zeile erhalten können, die durch den `DynamicRecord` dargestellt wird, indem Sie den Datensatz selbst inspizieren. Ein `DynamicRecord` ist vergleichbar mit einer `Row` in Apache Spark.

Def addField

```
def addField( path : String,
              dynamicNode : DynamicNode
            ) : Unit
```

Fügt einen DynamicNode zum angegebenen Pfad hinzu.

- `path` – Der Pfad für das hinzuzufügende Feld.
- `dynamicNode` – Die DynamicNode, der an dem angegebenen Pfad hinzugefügt werden soll.

Def dropField

```
def dropField(path: String, underRename: Boolean = false): Option[DynamicNode]
```

Verwirft ein DynamicNode im angegebenen Pfad und gibt den verworfenen Knoten zurück, wenn kein Array im angegebenen Pfad vorhanden ist.

- `path` – Der Pfad zum zu verwerfenden Feld.
- `underRenamedropField` – "True" wenn im Rahmen einer Umbenennungstransformationen aufgerufen wird, andernfalls "false" (Standard: "false").

Gibt eine `scala.Option Option` zurück (DynamicNode).

Def setError

```
def setError( error : Error )
```

Legt diesen Datensatz als Fehlerdatensatz fest, wie im `error`-Parameter angegeben.

Gibt eine `DynamicRecord` zurück.

Def isError

```
def isError
```

Prüft, ob der Datensatz ein Fehlerdatensatz ist.

Def getError

```
def getError
```

Ruft den `Error` ab, wenn der Datensatz ein Fehlerdatensatz ist. Gibt `scala.Some` Some (Fehler) zurück, wenn der Datensatz ein Fehlerdatensatz ist, andernfalls `scala.None`.

Def clearError

```
def clearError
```

Legen Sie den Wert für `Error` auf `scala.None.None` fest.

Def write

```
override def write( out : DataOutput ) : Unit
```

Def readFields

```
override def readFields( in : DataInput ) : Unit
```

Def clone

```
override def clone : DynamicRecord
```

Klont diesen Datensatz zu einem neuen `DynamicRecord` und gibt diesen zurück.

Def schema

```
def schema
```

Ruft durch Untersuchung des Datensatzes das `Schema` ab.

Def getRoot

```
def getRoot : ObjectNode
```

Liefert den Wurzel-`ObjectNode` für den Datensatz.

Def toJson

```
def toJson : String
```

Ruft die JSON-Zeichenfolge für den Datensatz ab.

Def getFieldNode

```
def getFieldNode( path : String ) : Option[DynamicNode]
```

Liefert den Wert des Feldes am angegebenen `path` als Option von `DynamicNode`.

Gibt `scala.Some` Some zurück (DynamicNode), wenn das Feld vorhanden ist, andernfalls `scala.None.None` .

Def getField

```
def getField( path : String ) : Option[Any]
```

Liefert den Wert des Feldes am angegebenen `path` als Option von `DynamicNode`.

Gibt `scala.Some` Some zurück (Wert).

Def hashCode

```
override def hashCode : Int
```

Def equals

```
override def equals( other : Any )
```

DynamicRecord Objekt

```
object DynamicRecord
```

Def apply

```
def apply( row : Row,
```

```
            schema : SparkStructType )
```

Wenden Sie die Methode an, um eine Apache Spark SQL-Row in einen DynamicRecord zu konvertieren.

- `row` – Eine Spark SQL-Row.
- `schema` – Das `Schema` dieser Zeile.

Gibt eine `DynamicRecord` zurück.

RecordTraverser Merkmal

```
trait RecordTraverser {
  def nullValue(): Unit
  def byteValue(value: Byte): Unit
  def binaryValue(value: Array[Byte]): Unit
  def booleanValue(value: Boolean): Unit
  def shortValue(value: Short) : Unit
  def intValue(value: Int) : Unit
  def longValue(value: Long) : Unit
  def floatValue(value: Float): Unit
  def doubleValue(value: Double): Unit
  def decimalValue(value: BigDecimal): Unit
  def stringValue(value: String): Unit
  def dateValue(value: Date): Unit
  def timestampValue(value: Timestamp): Unit
  def objectStart(length: Int): Unit
  def objectKey(key: String): Unit
  def objectEnd(): Unit
  def mapStart(length: Int): Unit
  def mapKey(key: String): Unit
  def mapEnd(): Unit
  def arrayStart(length: Int): Unit
  def arrayEnd(): Unit
}
```

AWS Glue Scala GlueContext APIs

Paket: com.amazonaws.services.glue

```
class GlueContext extends SQLContext(sc) (
           @transient val sc : SparkContext,
```

```
            val defaultSourcePartitioner : PartitioningStrategy )
```

`GlueContext` ist der Eintrittspunkt für das Lesen und Schreiben eines DynamicFrame von und zu Amazon Simple Storage Service (Amazon S3), den AWS Glue Data Catalog, JDBC und so weiter. Diese Klasse bietet Hilfsfunktionen zum Erstellen von DataSource Merkmal- und DataSink-Objekten, die wiederum zum Lesen und Schreiben von `DynamicFrames` verwendet werden können.

Sie können `GlueContext` auch verwenden, um eine Zielanzahl an Partitionen (Standard 20) im `DynamicFrame` festzulegen, wenn die Anzahl der von der Quelle erstellten Partitionen kleiner ist als ein Mindestwert für Partitionen (Standard 10).

Spalten definieren addIngestionTime

```
def addIngestionTimeColumns(
         df : DataFrame,
         timeGranularity : String = "") : dataFrame
```

Hängt die Erfassungszeitspalten wie `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` an die Eingabe `DataFrame` an. Diese Funktion wird automatisch in dem Skript generiert, das von AWS Glue wenn Sie eine Datenkatalogtabelle mit Amazon S3 als Ziel angeben. Diese Funktion aktualisiert automatisch die Partition mit Erfassungszeitspalten in der Ausgabetabelle. So können die Ausgabedaten bei der Erfassung automatisch partitioniert werden, ohne dass explizite Erfassungszeitspalten in den Eingabedaten erforderlich sind.

- `dataFrame` – Der `dataFrame`, um die Erfassungszeitspalten anzuhängen.
- `timeGranularity` – Die Granularität der Zeitspalten. Gültige Werte sind „`day`“, „`hour`“ und „`minute`“. Wenn zum Beispiel „`hour`“ an die Funktion übergeben wird, werden im Original `dataFrame` die Zeiten in den Spalten „`ingest_year`“, „`ingest_month`“, „`ingest_day`“, und „`ingest_hour`“ aktualisiert.

Gibt den Datenrahmen nach dem Anhängen der Zeitgranularitätsspalten zurück.

Beispiel:

```
glueContext.addIngestionTimeColumns(dataFrame, "hour")
```

def createDataFrame FromOptions

```
def createDataFrameFromOptions( connectionType : String,
```

```
                        connectionOptions : JsonOptions,
                        transformationContext : String = "",
                        format : String = null,
                        formatOptions : JsonOptions = JsonOptions.empty
                      ) : DataSource
```

Gibt einen `DataFrame` zurück, der mit der angegebenen Verbindung und dem Format erstellt wurde. Verwenden Sie diese Funktion nur mit AWS Glue-Streaming-Quellen.

- `connectionType` – Die Art der Streaming-Verbindung. Gültige Werte sind `kinesis` und `kafka`.
- `connectionOptions` – Verbindungsoptionen, die für Kinesis und Kafka unterschiedlich sind. Die Liste aller Verbindungsoptionen für jede Streaming-Datenquelle finden Sie unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark. Beachten Sie die folgenden Unterschiede bei den Streaming-Verbindungsoptionen:
  - Kinesis-Streaming-Quellen erfordern `streamARN`, `startingPosition`, `inferSchema` und `classification`.
  - Kafka-Streaming-Quellen erfordern `connectionName`, `topicName`, `startingOffsets`, `inferSchema` und `classification`.
- `transformationContext` – Der zu verwendende Transformationskontext (optional).
- `format` – Eine Formatspezifikation (optional). Dies wird für einen Amazon S3 oder einen verwendet AWS Glue Verbindung, die mehrere Formate unterstützt. Informationen zu den unterstützten Formaten finden Sie unter Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark
- `formatOptions` – Formatierungsoptionen für das angegebene Format. Weitere Informationen zu unterstützten Formatoptionen finden Sie unter Pfad-Formatoptionen.

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
val data_frame_datasource0 =
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kinesis",
connectionOptions = JsonOptions("""{"streamName": "example_stream", "startingPosition":
 "TRIM_HORIZON", "inferSchema": "true", "classification": "json"}}"""))
```

Beispiel für die Kafka-Streaming-Quelle:

```
val data_frame_datasource0 =
```

```
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kafka",
connectionOptions = JsonOptions("""{"connectionName": "example_connection",
 "topicName": "example_topic", "startingPosition": "earliest", "inferSchema": "false",
 "classification": "json", "schema":"`column1` STRING, `column2` STRING"}"""))
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

Wendet die `batch_function` auf jeden Mikrobatch an, der von der Streaming-Quelle gelesen wird.

- `frame`— Der DataFrame , der den aktuellen Mikrostapel enthält.
- `batch_function` – Eine Funktion, die für jeden Mikrobatch angewendet wird.
- `options` – Eine Sammlung von Schlüssel-Wert-Paaren, die Informationen zur Verarbeitung von Mikrobatches enthält. Die folgenden Optionen sind erforderlich:
  - `windowSize` – Die Zeitspanne für die Verarbeitung der einzelnen Batches.
  - `checkpointLocation` – Der Ort, an dem Checkpoints für den Streaming-ETL-Auftrag gespeichert werden.
  - `batchMaxRetries` – Die maximale Anzahl der Wiederholungsversuche für diesen Batch, wenn er fehlschlägt. Der Standardwert ist 3. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.

Beispiel:

```
glueContext.forEachBatch(data_frame_datasource0, (dataFrame: Dataset[Row], batchId:
 Long) =>
   {
      if (dataFrame.count() > 0)
        {
          val datasource0 = DynamicFrame(glueContext.addIngestionTimeColumns(dataFrame,
 "hour"), glueContext)
          // @type: DataSink
          // @args: [database = "tempdb", table_name = "fromoptionsoutput",
 stream_batch_time = "100 seconds",
          //      stream_checkpoint_location = "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/",
          //      transformation_ctx = "datasink1"]
          // @return: datasink1
          // @inputs: [frame = datasource0]
          val options_datasink1 = JsonOptions(
```

```
            Map("partitionKeys" -> Seq("ingest_year", "ingest_month","ingest_day",
 "ingest_hour"),
            "enableUpdateCatalog" -> true))
          val datasink1 = glueContext.getCatalogSink(
            database = "tempdb",
            tableName = "fromoptionsoutput",
            redshiftTmpDir = "",
            transformationContext = "datasink1",
            additionalOptions = options_datasink1).writeDynamicFrame(datasource0)
        }
   }, JsonOptions("""{"windowSize" : "100 seconds",
         "checkpointLocation" : "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/"}"""))
```

def getCatalogSink

```
def getCatalogSink( database : String,
        tableName : String,
        redshiftTmpDir : String = "",
        transformationContext : String = ""
        additionalOptions: JsonOptions = JsonOptions.empty,
        catalogId: String = null
) : DataSink
```

Erstellt eine DataSink, die in einen Speicherort schreibt, der in einer Tabelle angegeben ist, die im Data Catalog definiert ist.

- `database` – Der Datenbankname im Data Catalog.
- `tableName` – Der Tabellenname im Data Catalog.
- `redshiftTmpDir` – Das vorläufige Staging-Verzeichnis für die Verwendung mit bestimmten Datensenken. Standardmäßig auf „leer“ festgelegt.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `additionalOptions`— Zusätzliche Optionen stehen zur Verfügung AWS Glue.
- `catalogId` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei null wird die Standard-Konto-ID des Aufrufers verwendet.

Gibt den `DataSink` zurück.

def getCatalogSource

```
def getCatalogSource( database : String,
                      tableName : String,
                      redshiftTmpDir : String = "",
                      transformationContext : String = ""
                      pushDownPredicate : String = " "
                      additionalOptions: JsonOptions = JsonOptions.empty,
                      catalogId: String = null
                    ) : DataSource
```

Erstellt einen DataSource Merkmal, der Daten aus einer Tabellendefinition im Data Catalog liest.

- `database` – Der Datenbankname im Data Catalog.
- `tableName` – Der Tabellenname im Data Catalog.
- `redshiftTmpDir` – Das vorläufige Staging-Verzeichnis für die Verwendung mit bestimmten Datensenken. Standardmäßig auf „leer“ festgelegt.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `pushDownPredicate` – Filtert Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen. Weitere Informationen finden Sie unter Vorabfilterung mit Pushdown-Prädikaten.
- `additionalOptions` – Eine Sammlung optionaler Name/Wert-Paare. Zu den möglichen Optionen gehören die unter Verbindungsarten und Optionen für ETL in AWS Glue für Spark aufgeführten, außer `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification` und `delimiter`. Eine weitere unterstützte Option ist `catalogPartitionPredicate`:

  `catalogPartitionPredicate` – Sie können einen Katalogausdruck basierend auf den Indexspalten an den Filter übergeben. Dies verlagert die Filterung auf die Serverseite. Weitere Informationen finden Sie unter AWS Glue Partitionsindizes. Beachten Sie, dass `push_down_predicate` und `catalogPartitionPredicate` verschiedene Syntaxen verwenden. Erstere verwendet die Spark-SQL-Standardsyntax und letztere verwendet den JSQL-Parser.
- `catalogId` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei null wird die Standard-Konto-ID des Aufrufers verwendet.

Gibt den `DataSource` zurück.

Beispiel für eine Streaming-Quelle

```
val data_frame_datasource0 = glueContext.getCatalogSource(
    database = "tempdb",
    tableName = "test-stream-input",
    redshiftTmpDir = "",
    transformationContext = "datasource0",
    additionalOptions = JsonOptions("""{
        "startingPosition": "TRIM_HORIZON", "inferSchema": "false"}""")
    ).getDataFrame()
```

def abrufen JDBCSink

```
def getJDBCSink( catalogConnection : String,
                 options : JsonOptions,
                 redshiftTmpDir : String = "",
                 transformationContext : String = "",
                 catalogId: String = null
               ) : DataSink
```

Erstellt eine DataSink, die in eine JDBC-Datenbank schreibt, die im Data Catalog in einem `Connection`-Objekt angegeben ist. Das `Connection`-Objekt enthält Informationen zum Herstellen einer Verbindung zu einer JDBC-Senke, einschließlich URL, Benutzername, Passwort, VPC, Subnetz und Sicherheitsgruppen.

- `catalogConnection` – Der Name der Verbindung im Data Catalog, die die JDBC-URL zum Schreiben enthält.
- `options` – Eine Reihe von JSON-Namen-Wert-Paaren, die zusätzliche Informationen liefern, die zum Schreiben in einen JDBC-Datenspeicher erforderlich sind. Dies umfasst:
  - dbtable (erforderlich) – Der Name der JDBC-Tabelle. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie `schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet. Das folgende Beispiel zeigt einen Optionsparameter, der auf ein Schema mit Namen `test` und einer Tabelle mit Namen `test_table` in der Datenbank `test_db` verweist.

    ```
    options = JsonOptions("""{"dbtable": "test.test_table", "database": "test_db"}""")
    ```

  - database (erforderlich) – Der Name der JDBC-Datenbank.

  - Alle zusätzlichen Optionen werden direkt an den SparkSQL JDBC-Writer übergeben. Weitere Informationen finden Sie unter Redshift-Datenquelle für Spark.
- `redshiftTmpDir` – Ein vorläufiges Staging-Verzeichnis für die Verwendung mit bestimmten Datensenken. Standardmäßig auf „leer“ festgelegt.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `catalogId` – Die Katalog-ID (Konto-ID) des Data Catalogs, auf den zugegriffen wird. Bei null wird die Standard-Konto-ID des Aufrufers verwendet.

Beispiel-Code:

```
getJDBCSink(catalogConnection = "my-connection-name", options =
 JsonOptions("""{"dbtable": "my-jdbc-table", "database": "my-jdbc-db"}"""),
 redshiftTmpDir = "", transformationContext = "datasink4")
```

Gibt den `DataSink` zurück.

def getSink

```
def getSink( connectionType : String,
             connectionOptions : JsonOptions,
             transformationContext : String = ""
           ) : DataSink
```

Erstellt eineDataSink, die Daten in ein Ziel wie Amazon Simple Storage Service (Amazon S3), JDBC oder den AWS Glue Data Catalog oder einen Apache Kafka- oder Amazon Kinesis Kinesis-Datenstream schreibt.

- `connectionType` – Typ der Verbindung. Siehe the section called “Verbindungsparameter”.
- `connectionOptions` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Informationen zum Herstellen einer Verbindung mit der Datensenke bereitstellt. Siehe the section called “Verbindungsparameter”.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.

Gibt den `DataSink` zurück.

def-Format getSinkWith

```
def getSinkWithFormat( connectionType : String,
                       options : JsonOptions,
                       transformationContext : String = "",
                       format : String = null,
                       formatOptions : JsonOptions = JsonOptions.empty
                     ) : DataSink
```

Erstellt eineDataSink, die Daten in ein Ziel wie Amazon S3, JDBC oder den Datenkatalog oder einen Apache Kafka- oder Amazon Kinesis Kinesis-Datenstream schreibt. Legt auch das Format für die Daten fest, die an das Ziel geschrieben werden sollen.

- `connectionType` – Typ der Verbindung. Siehe the section called “Verbindungsparameter”.
- `options` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Informationen zum Herstellen einer Verbindung mit der Datensenke bereitstellt. Siehe the section called “Verbindungsparameter”.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `format` – Das Format der Daten, die in das Ziel geschrieben werden.
- `formatOptions` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Optionen zum Formatieren der Daten am Ziel bereitstellt. Siehe Pfad-Formatoptionen.

Gibt den `DataSink` zurück.

def getSource

```
def getSource( connectionType : String,
               connectionOptions : JsonOptions,
               transformationContext : String = ""
               pushDownPredicate
             ) : DataSource
```

Erstellt eineDataSource Merkmal, die Daten aus einer Quelle wie Amazon S3, JDBC oder dem AWS Glue-Datenkatalog liest. Unterstützt auch Kafka- und Kinesis-Streaming-Datenquellen.

- `connectionType` – Der Typ der Datenquelle. Siehe the section called “Verbindungsparameter”.

- `connectionOptions` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Informationen zum Herstellen einer Verbindung mit der Datenquelle bereitstellt. Weitere Informationen finden Sie unter the section called “Verbindungsparameter”.

  Für eine Kinesis-Streaming-Quelle sind die folgenden Verbindungsoptionen erforderlich: `streamARN`, `startingPosition`, `inferSchema` und `classification`.

  Für eine Kafka-Streaming-Quelle sind die folgenden Verbindungsoptionen erforderlich: `connectionName`, `topicName`, `startingOffsets`, `inferSchema` und `classification`.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `pushDownPredicate` – Prädikat für Partitionsspalten.

Gibt den `DataSource` zurück.

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
val kinesisOptions = jsonOptions()
data_frame_datasource0 = glueContext.getSource("kinesis",
 kinesisOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"streamARN": "arn:aws:kinesis:eu-central-1:123456789012:stream/
fromOptionsStream",
         |"startingPosition": "TRIM_HORIZON",
         |"inferSchema": "true",
         |"classification": "json"}""".stripMargin)
}
```

Beispiel für die Kafka-Streaming-Quelle:

```
val kafkaOptions = jsonOptions()
val data_frame_datasource0 = glueContext.getSource("kafka",
 kafkaOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"connectionName": "ConfluentKafka",
         |"topicName": "kafka-auth-topic",
```

```
        |"startingOffsets": "earliest",
        |"inferSchema": "true",
        |"classification": "json"}""".stripMargin)
 }
```

def-Format getSourceWith

```
def getSourceWithFormat( connectionType : String,
                         options : JsonOptions,
                         transformationContext : String = "",
                         format : String = null,
                         formatOptions : JsonOptions = JsonOptions.empty
                       ) : DataSource
```

Erstellt eineDataSource Merkmal, die Daten aus einer Quelle wie Amazon S3, JDBC oder dem AWS Glue-Datenkatalog liest und auch das Format der in der Quelle gespeicherten Daten festlegt.

- `connectionType` – Der Typ der Datenquelle. Siehe the section called “Verbindungsparameter”.
- `options` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Informationen zum Herstellen einer Verbindung mit der Datenquelle bereitstellt. Siehe the section called “Verbindungsparameter”.
- `transformationContext` – Der Transformationskontext, der mit der Senke verbunden ist, die von Auftrags-Lesezeichen zu verwenden ist. Standardmäßig auf „leer“ festgelegt.
- `format` – Das Format der in der Quelle gespeicherten Daten. Wenn der `connectionType` „s3“ ist, können Sie auch `format` angeben. Mögliche Werte sind: „avro“, „csv“, „grokLog“, „ion“, „json“, „xml“, „parquet“ oder „orc“.
- `formatOptions` – Eine Zeichenfolge eines JSON-Name-Wert-Paares, die zusätzliche Optionen zum Analysieren von Daten an der Quelle bereitstellt. Siehe Pfad-Formatoptionen.

Gibt den `DataSource` zurück.

Beispiele

Erstellen Sie eine Datei DynamicFrame aus einer Datenquelle, bei der es sich um eine Datei mit kommagetrennten Werten (CSV) auf Amazon S3 handelt:

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="s3",
```

```
    options =JsonOptions(s"""{"paths": [ "s3://csv/nycflights.csv"]}"""),
    transformationContext = "datasource0",
    format = "csv",
    formatOptions=JsonOptions(s"""{"withHeader":"true","separator": ","}""")
    ).getDynamicFrame()
```

Erstellen Sie mithilfe einer JDBC-Verbindung DynamicFrame aus einer Datenquelle, bei der es sich um eine PostgreSQL-Datenquelle handelt:

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="postgresql",
    options =JsonOptions(s"""{
      "url":"jdbc:postgresql://databasePostgres-1.rds.amazonaws.com:5432/testdb",
      "dbtable": "public.company",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

Erstellen Sie mithilfe einer JDBC-Verbindung DynamicFrame aus einer Datenquelle, bei der es sich um ein MySQL handelt:

```
 val datasource0 = glueContext.getSourceWithFormat(
    connectionType="mysql",
    options =JsonOptions(s"""{
      "url":"jdbc:mysql://databaseMysql-1.rds.amazonaws.com:3306/testdb",
      "dbtable": "athenatest_nycflights13_csv",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

def getSparkSession

```
def getSparkSession : SparkSession
```

Ruft das `SparkSession`-Objekt ab, das mit diesem GlueContext verknüpft ist. Verwenden Sie dieses SparkSession Objekt zur Registrierung von Tabellen und UDFs zur Verwendung mit `DataFrame` created from DynamicFrames.

Gibt das zurück SparkSession.

def startTransaction

```
def startTransaction(readOnly: Boolean):String
```

Starten Sie eine neue Transaktion. Ruft intern die startTransaction-API von Lake Formation auf.

- `readOnly` – (Boolean) Gibt an, ob diese Transaktion schreibgeschützt oder gelesen und geschrieben werden soll. Schreibvorgänge mit einer schreibgeschützten Transaktions-ID werden abgelehnt. Schreibgeschützte Transaktionen müssen nicht festgeschrieben werden.

Gibt die Transaktions-ID zurück.

def commitTransaction

```
def commitTransaction(transactionId: String, waitForCommit: Boolean): Boolean
```

Versucht, die angegebene Transaktion zu übernehmen. `commitTransaction` kann zurückkehren, bevor die Transaktion abgeschlossen ist. Ruft intern die CommitTransaction-API von Lake Formation auf.

- `transactionId` – (String) Die zu verbindende Transaktion.
- `waitForCommit` – (Boolean) Bestimmt, ob die Rückgabe von `commitTransaction` sofort erfolgt. Der Standardwert ist "True". Wenn false (falsch), befragt `commitTransaction` und wartet, bis die Transaktion übergeben wurde. Die Dauer der Wartezeit ist mit einem exponentiellen Backoff mit maximal 6 Wiederholungsversuchen auf 1 Minute beschränkt.

Gibt einen booleschen Wert zurück, um anzugeben, ob das Commit abgeschlossen ist oder nicht.

def cancelTransaction

```
def cancelTransaction(transactionId: String): Unit
```

Versucht, die angegebene Transaktion abzubrechen. Ruft intern die Lake Formation CancelTransactionAPI auf.

- `transactionId` – (String) Die abzubrechende Transaktion.

Gibt eine `TransactionCommittedException`-Ausnahme zurück, wenn die Transaktion zuvor festgeschrieben wurde.

def this

```
def this( sc : SparkContext,
          minPartitions : Int,
          targetPartitions : Int )
```

Erzeugt ein `GlueContext`-Objekt mit dem angegebenen `SparkContext`, minimalen Partitionen und Zielpartitionen.

- `sc` — Das Tool `SparkContext`.
- `minPartitions` – Die Mindestanzahl an Partitionen.
- `targetPartitions` – Die Zielanzahl an Partitionen.

Gibt den `GlueContext` zurück.

def this

```
def this( sc : SparkContext )
```

Erstellt ein `GlueContext`-Objekt mit dem mitgelieferten `SparkContext`. Legt die Mindestpartitionen auf 10 und die Zielpartitionen auf 20 fest.

- `sc` — Das Tool `SparkContext`.

Gibt den `GlueContext` zurück.

def this

```
def this( sparkContext : JavaSparkContext )
```

Erstellt ein `GlueContext`-Objekt mit dem mitgelieferten `JavaSparkContext`. Legt die Mindestpartitionen auf 10 und die Zielpartitionen auf 20 fest.

- `sparkContext` — Das Tool `JavaSparkContext`.

Gibt den `GlueContext` zurück.

MappingSpec

Paket: com.amazonaws.services.glue

MappingSpec Fallklasse

```
case class MappingSpec( sourcePath: SchemaPath,
                        sourceType: DataType,
                        targetPath: SchemaPath,
                        targetTyp
                       ) extends Product4[String, String, String, String] {
  override def _1: String = sourcePath.toString
  override def _2: String = ExtendedTypeName.fromDataType(sourceType)
  override def _3: String = targetPath.toString
  override def _4: String = ExtendedTypeName.fromDataType(targetType)
}
```

- `sourcePath` – Der `SchemaPath` des Quellfeldes.
- `sourceType` – Der `DataType` des Quellfeldes.
- `targetPath` – Der `SchemaPath` des Zielfelds.
- `targetType` – Der `DataType` des Zielfelds.

Ein `MappingSpec` gibt eine Zuweisung von einem Quellpfad und einem Quelldatentyp auf einen Zielpfad und einen Zieldatentyp an. Der Wert des Quellpfads im Quell-Frame wird im Ziel-Frame als Zielpfad angezeigt. Der Quelldatentyp wird in den Zieldatentyp umgewandelt.

Sie erbt von `Product4`, sodass Sie jedes beliebige `Product4` in Ihrer `applyMapping`-Schnittstelle verarbeiten können.

MappingSpec Objekt

```
object MappingSpec
```

Das `MappingSpec`-Objekt hat die folgenden Member:

Val orderingByTarget

```
val orderingByTarget: Ordering[MappingSpec]
```

Def apply

```
def apply( sourcePath : String,
           sourceType : DataType,
           targetPath : String,
           targetType : DataType
         ) : MappingSpec
```

Erzeugt einen `MappingSpec`.

- `sourcePath` – Eine Zeichenfolgendarstellung des Quellpfades.
- `sourceType` – Der `DataType` der Quelle.
- `targetPath` – Eine Zeichenfolgendarstellung des Zielpfads.
- `targetType` – Der `DataType` des Ziels.

Gibt eine `MappingSpec` zurück.

Def apply

```
def apply( sourcePath : String,
           sourceTypeString : String,
           targetPath : String,
           targetTypeString : String
         ) : MappingSpec
```

Erzeugt einen `MappingSpec`.

- `sourcePath` – Eine Zeichenfolgendarstellung des Quellpfades.
- `sourceType` – Eine Zeichenfolgendarstellung des Quelldatentyps.
- `targetPath` – Eine Zeichenfolgendarstellung des Zielpfads.
- `targetType` – Eine Zeichenfolgendarstellung des Zieldatentyps.

Gibt ein zurück MappingSpec.

Def apply

```
def apply( product : Product4[String, String, String, String] ) : MappingSpec
```

Erzeugt einen `MappingSpec`.

- `product` – `Product4` des Quellpfads, Quelldatentyps, Zielpfads und Zieldatentyps.

Gibt eine `MappingSpec` zurück.

AWS Glue Scala ResolveSpec APIs

Themen

- ResolveSpec Objekt
- ResolveSpec Fallklasse

Paket: com.amazonaws.services.glue

ResolveSpec Objekt

ResolveSpec

```
object ResolveSpec
```

Def

```
def apply( path : String,
           action : String
         ) : ResolveSpec
```

Erzeugt einen `ResolveSpec`.

- `path` – Eine Zeichenfolgendarstellung des aufzulösenden Auswahlfelds.
- `action` – Eine Auflösungsaktion. Die Aktion kann eine der folgenden sein: `Project`, `KeepAsStruct` oder `Cast`.

Gibt den `ResolveSpec` zurück.

Def

```
def apply( product : Product2[String, String] ) : ResolveSpec
```

Erzeugt einen `ResolveSpec`.

- `product` – `Product2` von: Quellpfad, Auflösungsaktion.

Gibt den `ResolveSpec` zurück.

ResolveSpec Fallklasse

```
case class ResolveSpec extends Product2[String, String]  (
           path : SchemaPath,
           action : String )
```

Erzeugt einen `ResolveSpec`.

- `path` – Der `SchemaPath` des Auswahlfeldes, das aufgelöst werden soll.
- `action` – Eine Auflösungsaktion. Die Aktion kann eine der folgenden sein: `Project`, `KeepAsStruct` oder `Cast`.

ResolveSpec Methoden definieren

```
def _1 : String
```

```
def _2 : String
```

AWS Glue Scala ArrayNode APIs

Paket: com.amazonaws.services.glue.types

ArrayNode Fallklasse

ArrayNode

```
case class ArrayNode extends DynamicNode  (
           value : ArrayBuffer[DynamicNode] )
```

ArrayNode Methoden definieren

```
def add( node : DynamicNode )
```

```
def clone
```

```
def equals( other : Any )
```

```
def get( index : Int ) : Option[DynamicNode]
```

```
def getValue
```

```
def hashCode : Int
```

```
def isEmpty : Boolean
```

```
def nodeType
```

```
def remove( index : Int )
```

```
def this
```

```
def toIterator : Iterator[DynamicNode]
```

```
def toJson : String
```

```
def update( index : Int,
            node : DynamicNode )
```

AWS Glue Scala BinaryNode APIs

Paket: com.amazonaws.services.glue.types

BinaryNode Fallklasse

BinaryNode

```
case class BinaryNode extends ScalarNode(value, TypeCode.BINARY)  (
           value : Array[Byte] )
```

BinaryNode Felder vom Typ Val

- `ordering`

BinaryNode Methoden definieren

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

AWS Glue Scala BooleanNode APIs

Paket: com.amazonaws.services.glue.types

BooleanNode Fallklasse

BooleanNode

```
case class BooleanNode extends ScalarNode(value, TypeCode.BOOLEAN)  (
           value : Boolean )
```

BooleanNode Felder vom Typ Val

- `ordering`

BooleanNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala ByteNode APIs

Paket: com.amazonaws.services.glue.types

ByteNode Fallklasse

ByteNode

```
case class ByteNode extends ScalarNode(value, TypeCode.BYTE)  (
```

```
            value : Byte )
```

ByteNode Felder vom Typ Val

- `ordering`

ByteNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala DateNode APIs

Paket: com.amazonaws.services.glue.types

DateNode Fallklasse

DateNode

```
case class DateNode extends ScalarNode(value, TypeCode.DATE)  (
            value : Date )
```

DateNode Felder vom Typ Val

- `ordering`

DateNode Methoden definieren

```
def equals( other : Any )
```

```
def this( value : Int )
```

AWS Glue Scala DecimalNode APIs

Paket: com.amazonaws.services.glue.types

DecimalNode Fallklasse

DecimalNode

```
case class DecimalNode extends ScalarNode(value, TypeCode.DECIMAL)  (
```

```
            value : BigDecimal )
```

DecimalNode Felder vom Typ Val

- `ordering`

DecimalNode Methoden definieren

```
def equals( other : Any )
```

```
def this( value : Decimal )
```

AWS Glue Scala DoubleNode APIs

Paket: com.amazonaws.services.glue.types

DoubleNode Fallklasse

DoubleNode

```
case class DoubleNode extends ScalarNode(value, TypeCode.DOUBLE)  (
            value : Double )
```

DoubleNode Felder vom Typ Val

- `ordering`

DoubleNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala DynamicNode APIs

Themen

- DynamicNode Klasse
- DynamicNode Objekt

Paket: com.amazonaws.services.glue.types

DynamicNode Klasse

DynamicNode

```
class DynamicNode extends Serializable with Cloneable
```

DynamicNode Methoden definieren

```
def getValue : Any
```

Abrufen eines einfachen Werts und Binden an den aktuellen Datensatz:

```
def nodeType : TypeCode
```

```
def toJson : String
```

Methode zum Debuggen:

```
def toRow( schema : Schema,
           options : Map[String, ResolveOption]
         ) : Row
```

```
def typeName : String
```

DynamicNode Objekt

DynamicNode

```
object DynamicNode
```

DynamicNode Methoden definieren

```
def quote( field : String,
           useQuotes : Boolean
         ) : String
```

```
def quote( node : DynamicNode,
           useQuotes : Boolean
         ) : String
```

EvaluateDataQuality Klasse

AWS Glue Data Quality befindet sich in der Vorschauversion für AWS Glue und kann sich ändern.

Paket: com.amazonaws.services.glue.dq

```
object EvaluateDataQuality
```

Def apply

```
def apply(frame: DynamicFrame,
            ruleset: String,
            publishingOptions: JsonOptions = JsonOptions.empty): DynamicFrame
```

Wertet einen Datenqualitätsregelsatz anhand eines `DynamicFrame` aus und gibt ein neues `DynamicFrame` mit den Ergebnissen der Bewertung zurück. Weitere Informationen über AWS Glue Data Quality finden Sie unterAWS Glue Qualität der Daten.

- `frame` – Das `DynamicFrame`, dessen Datenqualität Sie auswerten möchten.
- `ruleset` – Ein Regelsatz der Definitionssprache für Datenqualität (DQDL) im Zeichenfolgen-Format. Weitere Informationen zu DQDL finden Sie im Referenz zu Data Quality Definition Language (DQDL)-Benutzerhandbuch.
- `publishingOptions` – Ein Wörterbuch, das die folgenden Optionen zum Veröffentlichen von Auswertungsergebnissen und -metriken festlegt:
  - `dataQualityEvaluationContext`— Eine Zeichenfolge, die den Namespace angibt, unter dem AWS Glue Amazon CloudWatch Metriken und die Datenqualitätsergebnisse veröffentlichen soll. Die aggregierten Metriken werden in der Glue Studio-Oberfläche angezeigt CloudWatch, während die vollständigen Ergebnisse in der Benutzeroberfläche von AWS Glue Studio angezeigt werden.
    - Erforderlich: Nein
    - Standardwert: `default_context`
  - `enableDataQualityCloudWatchMetrics`— Gibt an, ob die Ergebnisse der Datenqualitätsbewertung veröffentlicht werden sollen. CloudWatch Mit der `dataQualityEvaluationContext`-Option geben Sie einen Namespace für die Metriken an.
    - Erforderlich: Nein

  - Standardwert: false
- `enableDataQualityResultsPublishing` – Gibt an, ob die Datenqualitätsergebnisse auf der Registerkarte Data Quality (Datenqualität) in der Benutzeroberfläche von AWS Glue Studio angezeigt werden sollen.
  - Erforderlich: Nein
  - Standardwert: wahr
- `resultsS3Prefix`— Gibt den Amazon S3 S3-Speicherort an, an den AWS Glue die Ergebnisse der Datenqualitätsbewertung schreiben kann.
  - Erforderlich: Nein
  - Standardwert: "" (eine leere Zeichenfolge)

Beispiel

Der folgende Beispielcode zeigt, wie Sie die Datenqualität für ein `DynamicFrame` bewerten, bevor Sie eine `SelectFields`-Transformation durchführen. Das Skript überprüft, ob alle Datenqualitätsregeln erfolgreich sind, bevor es versucht, die Transformation durchzuführen.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.dq.EvaluateDataQuality

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Create DynamicFrame with data
    val Legislators_Area = glueContext.getCatalogSource(database="legislators",
 tableName="areas_json", transformationContext="S3bucket_node1").getDynamicFrame()
```

```
    // Define data quality ruleset
    val DQ_Ruleset = """
      Rules = [ColumnExists "id"]
    """

    // Evaluate data quality
    val DQ_Results = EvaluateDataQuality.apply(frame=Legislators_Area,
 ruleset=DQ_Ruleset, publishingOptions=JsonOptions("""{"dataQualityEvaluationContext":
 "Legislators_Area", "enableDataQualityMetrics": "true",
 "enableDataQualityResultsPublishing": "true"}"""))
    assert(DQ_Results.filter(_.getField("Outcome").contains("Failed")).count == 0,
 "Failing DQ rules for Legislators_Area caused the job to fail.")

    // Script generated for node Select Fields
    val SelectFields_Results = Legislators_Area.selectFields(paths=Seq("id", "name"),
 transformationContext="Legislators_Area")

    Job.commit()
  }
}
```

AWS Glue Scala FloatNode APIs

Paket: com.amazonaws.services.glue.types

FloatNode Fallklasse

FloatNode

```
case class FloatNode extends ScalarNode(value, TypeCode.FLOAT)  (
           value : Float )
```

FloatNode Felder vom Typ Val

- `ordering`

FloatNode Methoden definieren

```
def equals( other : Any )
```

FillMissingValues Klasse

Paket: com.amazonaws.services.glue.ml

```
object FillMissingValues
```

Def apply

```
def apply(frame: DynamicFrame,
          missingValuesColumn: String,
          outputColumn: String = "",
          transformationContext: String = "",
          callSite: CallSite = CallSite("Not provided", ""),
          stageThreshold: Long = 0,
          totalThreshold: Long = 0): DynamicFrame
```

Trägt die fehlenden Werte eines Dynamic Frames in eine angegebene Spalte ein und gibt einen neuen Frame mit Schätzungen in einer neuen Spalte zurück. Bei Zeilen ohne fehlende Werte wird der Wert der angegebenen Spalte in die neue Spalte dupliziert.

- `frame`— Die, DynamicFrame in die fehlende Werte gefüllt werden sollen. Erforderlich
- `missingValuesColumn` – Die Spalte, die fehlende Werte enthält (`null`-Werte und leere Zeichenfolgen). Erforderlich
- `outputColumn` – Der Name der neuen Spalte, die geschätzte Werte für alle Zeilen enthält, deren Wert gefehlt hat. Optional; der Standardwert ist der Wert von `missingValuesColumn` mit Suffix `"_filled"`.
- `transformationContext` – Eine eindeutige Zeichenfolge zur Identifikation von Statusinformationen (optional).
- `callSite` – Wird verwendet, um Kontextinformationen für Fehlerberichte bereitzustellen (optional).
- `stageThreshold` – Die maximale Anzahl von Fehlern, die in der Transformation auftreten dürfen, bevor der Vorgang abgebrochen wird (optional, Standardwert ist Null).
- `totalThreshold` – Die maximale Anzahl von Fehlern, die insgesamt auftreten dürfen, bevor die Verarbeitung abgebrochen wird (optional, Standardwert ist Null).

Gibt einen neuen Dynamic Frame mit einer zusätzlichen Spalte zurück, die Schätzungen für Zeilen mit fehlenden Werten und den aktuellen Wert für andere Zeilen enthält.

FindMatches Klasse

Paket: com.amazonaws.services.glue.ml

```
object FindMatches
```

Def apply

```
def apply(frame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Suchen Sie Übereinstimmungen in einem Eingabe-Frame und geben Sie einen neuen Frame mit einer neuen Spalte zurück, die eine eindeutige ID pro Übereinstimmungsgruppe enthält.

- `frame`— Die, DynamicFrame in der nach Übereinstimmungen gesucht werden kann. Erforderlich
- `transformId`— Eine eindeutige ID, die der FindMatches Transformation zugeordnet ist, die auf den Eingabe-Frame angewendet werden soll. Erforderlich
- `transformationContext` – Die Kennung für diesen `DynamicFrame`. Der `transformationContext` wird als Schlüssel für den Auftrags-Lesezeichenstatus verwendet, der während der Ausführungen persistent ist. Optional.
- `callSite` – Wird verwendet, um Kontextinformationen für Fehlerberichte bereitzustellen. Diese Werte werden automatisch beim Aufruf von Python festgelegt. Optional.
- `stageThreshold` – Die maximale Anzahl der Fehlerdatensätze, die aufgrund der Berechnung dieses `DynamicFrame` zulässig sind, ehe eine Ausnahme ausgelöst wird. Ausgenommen sind Datensätze aus dem vorherigen `DynamicFrame`. Optional. Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl der Gesamtfehlerdatensätze, bevor eine Ausnahme ausgelöst wird, einschließlich derjenigen aus früheren Frames. Optional. Der Standardwert ist „Null“.
- `enforcedMatches` – Der Rahmen für erzwungene Übereinstimmungen. Optional. Der Standardwert ist `null`.

- `computeMatchConfidenceScores` – Ein boolescher Wert, der angibt, ob für jede Gruppe übereinstimmender Datensätze ein Konfidenzwert berechnet werden soll. Optional. Der Standardwert lautet „false“.

Gibt einen neuen Dynamic Frame mit einer eindeutigen Kennung zurück, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

FindIncrementalMatches Klasse

Paket: com.amazonaws.services.glue.ml

```
object FindIncrementalMatches
```

Def apply

```
apply(existingFrame: DynamicFrame,
            incrementalFrame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Suchen Sie Übereinstimmungen in den vorhandenen und inkrementellen Frames und geben Sie einen neuen Frame mit einer Spalte zurück, die eine eindeutige ID pro Übereinstimmungsgruppe enthält.

- `existingframe` – Ein vorhandener Frame, dem für jede Gruppe eine passende ID zugewiesen wurde. Erforderlich
- `incrementalframe` – Ein inkrementeller Frame, der zum Suchen von Übereinstimmungen mit dem vorhandenen Frame verwendet wird. Erforderlich
- `transformId`— Eine eindeutige ID, die der FindIncrementalMatches Transformation zugeordnet ist und auf die Eingabe-Frames angewendet werden soll. Erforderlich
- `transformationContext` – Die Kennung für diesen `DynamicFrame`. Der `transformationContext` wird als Schlüssel für den Auftrags-Lesezeichenstatus verwendet, der während der Ausführungen persistent ist. Optional.

- `callSite` – Wird verwendet, um Kontextinformationen für Fehlerberichte bereitzustellen. Diese Werte werden automatisch beim Aufruf von Python festgelegt. Optional.
- `stageThreshold` – Die maximale Anzahl der Fehlerdatensätze, die aufgrund der Berechnung dieses `DynamicFrame` zulässig sind, ehe eine Ausnahme ausgelöst wird. Ausgenommen sind Datensätze aus dem vorherigen `DynamicFrame`. Optional. Der Standardwert ist „Null“.
- `totalThreshold` – Die maximale Anzahl der Gesamtfehlerdatensätze, bevor eine Ausnahme ausgelöst wird, einschließlich derjenigen aus früheren Frames. Optional. Der Standardwert ist „Null“.
- `enforcedMatches` – Der Rahmen für erzwungene Übereinstimmungen. Optional. Der Standardwert ist `null`.
- `computeMatchConfidenceScores` – Ein boolescher Wert, der angibt, ob für jede Gruppe übereinstimmender Datensätze ein Konfidenzwert berechnet werden soll. Optional. Der Standardwert lautet „false“.

Gibt einen neuen Dynamic Frame mit einer eindeutigen Kennung zurück, die jeder Gruppe übereinstimmender Datensätze zugewiesen ist.

AWS Glue Scala IntegerNode APIs

Paket: com.amazonaws.services.glue.types

IntegerNode Fallklasse

IntegerNode

```
case class IntegerNode extends ScalarNode(value, TypeCode.INT)  (
           value : Int )
```

IntegerNode Felder vom Typ Val

- `ordering`

IntegerNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala LongNode APIs

Paket: com.amazonaws.services.glue.types

LongNode Fallklasse

LongNode

```
case class LongNode extends ScalarNode(value, TypeCode.LONG)  (
           value : Long )
```

LongNode Felder vom Typ Val

- `ordering`

LongNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala MapLikeNode APIs

Paket: com.amazonaws.services.glue.types

MapLikeNode Klasse

MapLikeNode

```
class MapLikeNode extends DynamicNode  (
           value : mutable.Map[String, DynamicNode] )
```

MapLikeNode Methoden definieren

```
def clear : Unit
```

```
def get( name : String ) : Option[DynamicNode]
```

```
def getValue
```

```
def has( name : String ) : Boolean
```

```
def isEmpty : Boolean
```

```
def put( name : String,
         node : DynamicNode
       ) : Option[DynamicNode]
```

```
def remove( name : String ) : Option[DynamicNode]
```

```
def toIterator : Iterator[(String, DynamicNode)]
```

```
def toJson : String
```

```
def toJson( useQuotes : Boolean ) : String
```

Beispiel: Angesichts dieses JSONs:

```
{"foo": "bar"}
```

Wenn `useQuotes == true`, ergibt `toJson` `{"foo": "bar"}`. Wenn `useQuotes == false`, ergibt `toJson` `{foo: bar}` @return.

AWS Glue Scala MapNode APIs

Paket: com.amazonaws.services.glue.types

MapNode Fallklasse

MapNode

```
case class MapNode extends MapLikeNode(value)  (
           value : mutable.Map[String, DynamicNode] )
```

MapNode Methoden definieren

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS Glue Scala NullNode APIs

Themen

- NullNode Klasse
- NullNode Case-Objekt

Paket: com.amazonaws.services.glue.types

NullNode Klasse

NullNode

```
class NullNode
```

NullNode Case-Objekt

NullNode

```
case object NullNode extends NullNode
```

AWS Glue Scala ObjectNode APIs

Themen

- ObjectNode Objekt
- ObjectNode Fallklasse

Paket: com.amazonaws.services.glue.types

ObjectNode Objekt

ObjectNode

```
object ObjectNode
```

ObjectNode Methoden definieren

```
def apply( frameKeys : Set[String],
           v1 : mutable.Map[String, DynamicNode],
           v2 : mutable.Map[String, DynamicNode],
           resolveWith : String
         ) : ObjectNode
```

ObjectNode Fallklasse

ObjectNode

```
case class ObjectNode extends MapLikeNode(value)  (
           val value : mutable.Map[String, DynamicNode] )
```

ObjectNode Methoden definieren

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS Glue Scala ScalarNode APIs

Themen

- ScalarNode Klasse
- ScalarNode Objekt

Paket: com.amazonaws.services.glue.types

ScalarNode Klasse

ScalarNode

```
class ScalarNode extends DynamicNode  (
           value : Any,
           scalarType : TypeCode )
```

ScalarNode Methoden definieren

```
def compare( other : Any,
             operator : String
           ) : Boolean
```

```
def getValue
```

```
def hashCode : Int
```

```
def nodeType
```

```
def toJson
```

ScalarNode Objekt

ScalarNode

```
object ScalarNode
```

ScalarNode Methoden definieren

```
def apply( v : Any ) : DynamicNode
```

```
def compare( tv : Ordered[T],
             other : T,
             operator : String
           ) : Boolean
```

```
def compareAny( v : Any,
                y : Any,
                o : String )
```

```
def withEscapedSpecialCharacters( jsonToEscape : String ) : String
```

AWS Glue Scala ShortNode APIs

Paket: com.amazonaws.services.glue.types

ShortNode Fallklasse

ShortNode

```
case class ShortNode extends ScalarNode(value, TypeCode.SHORT)  (
           value : Short )
```

ShortNode Felder vom Typ Val

- `ordering`

ShortNode Methoden definieren

```
def equals( other : Any )
```

AWS Glue Scala StringNode APIs

Paket: com.amazonaws.services.glue.types

StringNode Fallklasse

StringNode

```
case class StringNode extends ScalarNode(value, TypeCode.STRING)  (
           value : String )
```

StringNode Felder vom Typ Val

- `ordering`

StringNode Methoden definieren

```
def equals( other : Any )
```

```
def this( value : UTF8String )
```

AWS Glue Scala TimestampNode APIs

Paket: com.amazonaws.services.glue.types

TimestampNode Fallklasse

TimestampNode

```
case class TimestampNode extends ScalarNode(value, TypeCode.TIMESTAMP)  (
           value : Timestamp )
```

TimestampNode Felder vom Typ Val

- `ordering`

TimestampNode Methoden definieren

```
def equals( other : Any )
```

```
def this( value : Long )
```

AWS Glue Scala GlueArgParser APIs

Package: com.amazonaws.services.glue.util

GlueArgParser-Objekt

GlueArgParser

```
object GlueArgParser
```

Dies steht im Einklang mit der Python-Version von `utils.getResolvedOptions` im Paket `AWSGlueDataplanePython`.

GlueArgParser Methoden definieren

```
def getResolvedOptions( args : Array[String],
                        options : Array[String]
                      ) : Map[String, String]
```

```
def initParser( userOptionsSet : mutable.Set[String] ) : ArgumentParser
```

Example Abrufen der an einen Auftrag übergebenen Argumente

Zum Abrufen von Auftragsargumenten können Sie die `getResolvedOptions`-Methode verwenden. Betrachten Sie das folgende Beispiel, das ein Auftragsargument mit dem Namen `aws_region` abruft.

```
val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME","aws_region").toArray)
Job.init(args("JOB_NAME"), glueContext, args.asJava)
val region = args("aws_region")
println(region)
```

AWS Glue Job bei Scala APIs

Package: com.amazonaws.services.glue.util

Auftragsobjekt

Job

```
object Job
```

Job def-Methoden

```
def commit
```

```
def init( jobName : String,
          glueContext : GlueContext,
          args : java.util.Map[String, String] = Map[String, String]().asJava
        ) : this.type
```

```
def init( jobName : String,
          glueContext : GlueContext,
          endpoint : String,
          args : java.util.Map[String, String]
        ) : this.type
```

```
def isInitialized
```

```
def reset
```

```
def runId
```

Funktionen und Optimierungen für die Programmierung AWS Glue für Spark-ETL-Skripte

In den folgenden Abschnitten werden Techniken und Werte beschrieben, die allgemein gelten für AWS Glue für Spark-ETL-Programmierung (Extrahieren, Transformieren und Laden) in einer beliebigen Sprache.

Themen

- Verbindungsarten und Optionen für ETL in AWS Glue für Spark
- Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark
- AWS Glue Data Catalog-Unterstützung für Spark-SQL-Jobs
- Verwenden von Auftragslesezeichen
- Erkennung sensibler Daten außerhalb von AWS Glue Studio verwenden
- AWS Glue Visual Job API

Verbindungsarten und Optionen für ETL in AWS Glue für Spark

In AWS Glue Für Spark spezifizieren verschiedene Methoden PySpark und Transformationen sowie Scala-Methoden den Verbindungstyp mithilfe eines `connectionType` Parameters. Sie geben Verbindungsoptionen mit einem `connectionOptions`- oder `options`-Parameter an.

Der `connectionType`-Parameter kann die in der folgenden Tabelle angegebenen Werte annehmen. Die zugehörigen `connectionOptions`- (oder `options`)-Parameterwerte für jeden Typ sind in den

folgenden Abschnitten dokumentiert. Sofern nicht anders angegeben, gelten die Parameter, wenn die Verbindung als Quelle oder Senke verwendet wird.

Beispielcode, der das Festlegen und Verwenden von Verbindungsoptionen veranschaulicht, finden Sie auf der Homepage für jeden Verbindungstyp.

`connectionType`	Verbindet mit
dynamodb	Amazon-DynamoDB-Datenbank
kinesis	Amazon Kinesis Data Streams
S3	Amazon S3
documentdb	Amazon-DocumentDB-Datenbank (mit MongoDB-Kompatibilität)
opensearch	OpenSearch Amazon-Dienst.
redshift	Amazon Redshift-Datenbank
kafka	Kafka oder Amazon Managed Streaming for Apache Kafka
azurecosmos	Azure Cosmos for NoSQL.
azuresql	Azure SQL.
bigquery	Google BigQuery.
mongodb	MongoDB-Datenbank, einschließlich MongoDB Atlas.
sqlserver	Microsoft SQL Server-Datenbank (siehe JDBC-Verbindungen)
mysql-	MySQL-Datenbankserver (siehe JDBC-Verbindungen).
oracle	Oracle-Datenbank (siehe JDBC-Verbindungen)
postgresql	PostgreSQL-Datenbank (siehe JDBC-Verbindungen)
saphana	SAP HANA.
snowflake	Snowflake Data Lake

`connectionType`	Verbindet mit
teradata	Teradata Vantage.
vertica	Vertica.
benutzerdefiniert.*	Spark-, Athena- oder JDBC-Datenspeicher (siehe Custom- und AWS Marketplace ConnectionType-Werte
marketplace.*	Spark-, Athena- oder JDBC-Datenspeicher (siehe Custom- und AWS Marketplace ConnectionType-Werte)

DynamoDB-Verbindungen

Sie können AWS Glue for Spark verwenden, um in DynamoDB in Glue aus Tabellen zu lesen und in sie zu schreiben. AWS Sie stellen mithilfe der IAM-Berechtigungen, die mit Ihrem AWS Glue-Job verknüpft sind, eine Verbindung zu DynamoDB her. AWS Glue unterstützt das Schreiben von Daten in die DynamoDB-Tabelle eines anderen AWS Kontos. Weitere Informationen finden Sie unter the section called “Konten- und regionenübergreifender Zugriff auf DynamoDB-Tabellen”.

Zusätzlich zu AWS Glue DynamoDB-ETL-Konnektor, Sie können mithilfe des DynamoDB-Exportconnectors aus DynamoDB lesen, der eine `ExportTableToPointInTime` DynamoDB-Anfrage aufruft und sie an einem von Ihnen angegebenen Amazon S3 S3-Speicherort im Format von DynamoDB JSON speichert. AWS Glue erstellt dann ein DynamicFrame Objekt, indem die Daten vom Amazon S3 S3-Exportspeicherort gelesen werden.

Der DynamoDB-Writer ist verfügbar in AWS Glue Version 1.0 oder spätere Versionen. Das Tool AWS Glue Der DynamoDB-Exportconnector ist verfügbar in AWS Glue Version 2.0 oder spätere Versionen.

Weitere Informationen zu DynamoDB finden Sie in der Dokumentation zu Amazon DynamoDB.

> **Note**
>
> Der DynamoDB-ETL-Reader unterstützt keine Filter oder Pushdown-Prädikate.

Konfigurieren von DynamoDB-Verbindungen

Um von AWS Glue aus eine Verbindung zu DynamoDB herzustellen, erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle die Erlaubnis, mit DynamoDB zu interagieren. Weitere Informationen zu Berechtigungen auf Ressourcenebene, die zum Lesen und Schreiben in DynamoDB erforderlich sind, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für DynamoDB in der Dokumentation zu IAM.

In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

- Wenn Sie den DynamoDB-Export-Konnektor verwenden, müssen Sie IAM so konfigurieren, dass Ihr Auftrag DynamoDB-Tabellenexporte anfordern kann. Darüber hinaus müssen Sie einen Amazon S3 S3-Bucket für den Export identifizieren und die entsprechenden Berechtigungen in IAM bereitstellen, damit DynamoDB in diesen Bucket schreiben und Ihr AWS Glue-Job daraus lesen kann. Weitere Informationen finden Sie unter Anfordern eines Tabellenexports in DynamoDB.
- Wenn Ihr AWS Glue-Job spezielle Amazon VPC-Konnektivitätsanforderungen hat, verwenden Sie den `NETWORK` AWS Glue-Verbindungstyp, um Netzwerkoptionen bereitzustellen. Da der Zugriff auf DynamoDB von IAM autorisiert wird, ist kein AWS Glue DynamoDB-Verbindungstyp erforderlich.

Aus DynamoDB lesen und in DynamoDB schreiben

In den folgenden Codebeispielen wird gezeigt, wie DynamoDB-Tabellen (über den ETL Connector) gelesen und geschrieben werden. Sie demonstrieren das Lesen von einer Tabelle und das Schreiben in eine andere Tabelle.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
```

```
    connection_options={"dynamodb.input.tableName": test_source,
        "dynamodb.throughput.read.percent": "1.0",
        "dynamodb.splits": "100"
    }
)
print(dyf.getNumPartitions())

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={"dynamodb.output.tableName": test_sink,
        "dynamodb.throughput.write.percent": "1.0"
    }
)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.input.tableName" -> test_source,
        "dynamodb.throughput.read.percent" -> "1.0",
        "dynamodb.splits" -> "100"
      ))
    ).getDynamicFrame()
```

```
    print(dynamicFrame.getNumPartitions())

    val dynamoDbSink: DynamoDbDataSink =  glueContext.getSinkWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.output.tableName" -> test_sink,
        "dynamodb.throughput.write.percent" -> "1.0"
      ))
    ).asInstanceOf[DynamoDbDataSink]

    dynamoDbSink.writeDynamicFrame(dynamicFrame)

    Job.commit()
  }

}
```

Verwenden des DynamoDB-Export-Konnektors

Der Export-Konnektor funktioniert besser als der ETL-Konnektor, wenn die DynamoDB-Tabelle größer als 80 GB ist. Da die Exportanforderung außerdem außerhalb der Spark-Prozesse in einem AWS Glue Job, Sie können die auto Skalierung von AWS Glue-Jobs aktivieren, um die DPU-Nutzung während der Exportanforderung zu sparen. Mit dem Export-Konnektor müssen Sie auch nicht die Anzahl der Aufteilungen für die Parallelität des Spark-Executors oder den Prozentsatz des DynamoDB-Durchsatzes konfigurieren.

> **Note**
>
> DynamoDB hat spezifische Anforderungen, um `ExportTableToPointInTime`-Anforderungen aufzurufen. Weitere Informationen finden Sie unter Anfordern eines Tabellenexports in DynamoDB. Zum Beispiel muss Point-in-Time-Restore (PITR) in der Tabelle aktiviert sein, um diesen Konnektor verwenden zu können. Der DynamoDB-Connector unterstützt auch die AWS KMS Verschlüsselung von DynamoDB-Exporten nach Amazon S3. Wenn Sie Ihre Sicherheitskonfiguration in der AWS Glue-Job-Konfiguration angeben, wird die AWS KMS Verschlüsselung für einen DynamoDB-Export aktiviert. Der KMS-Schlüssel muss sich in derselben Region wie der Amazon-S3-Bucket befinden. Beachten Sie, dass zusätzliche Gebühren für den DynamoDB-Export und Amazon-S3-Speicherkosten anfallen. Exportierte Daten in Amazon S3 bleiben bestehen, nachdem

ein Auftrag ausgeführt wurde, sodass Sie sie ohne zusätzliche DynamoDB-Exporte wiederverwenden können. Voraussetzung für die Verwendung dieses Connectors ist, dass point-in-time Recovery (PITR) für die Tabelle aktiviert ist.
Der DynamoDB-ETL-Konnektor oder Export-Konnektor unterstützt keine Filter oder Pushdown-Prädikate, die an der DynamoDB-Quelle angewendet werden sollen.

In den folgenden Codebeispielen wird gezeigt, wie die Anzahl der Partitionen (über den Export-Konnektor) gelesen und gedruckt werden.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": test_source,
        "dynamodb.s3.bucket": bucket_name,
        "dynamodb.s3.prefix": bucket_prefix,
        "dynamodb.s3.bucketOwner": account_id_of_bucket,
    }
)
print(dyf.getNumPartitions())

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
```

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> test_source,
        "dynamodb.s3.bucket" -> bucket_name,
        "dynamodb.s3.prefix" -> bucket_prefix,
        "dynamodb.s3.bucketOwner" -> account_id_of_bucket,
      ))
    ).getDynamicFrame()

    print(dynamicFrame.getNumPartitions())

    Job.commit()
  }

}
```

Diese Beispiele zeigen, wie aus einer AWS Glue Data Catalog-Tabelle, die eine dynamodb Klassifizierung hat, gelesen (über den Export-Connector) und wie die Anzahl der Partitionen gedruckt wird:

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions
```

```
args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_catalog(
        database=catalog_database,
        table_name=catalog_table_name,
        additional_options={
            "dynamodb.export": "ddb",
            "dynamodb.s3.bucket": s3_bucket,
            "dynamodb.s3.prefix": s3_bucket_prefix
        }
    )
print(dynamicFrame.getNumPartitions())

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getCatalogSource(
        database = catalog_database,
        tableName = catalog_table_name,
        additionalOptions = JsonOptions(Map(
            "dynamodb.export" -> "ddb",
            "dynamodb.s3.bucket" -> s3_bucket,
```

```
            "dynamodb.s3.prefix" -> s3_bucket_prefix
        ))
    ).getDynamicFrame()
    print(dynamicFrame.getNumPartitions())
)
```

Vereinfachung der Verwendung einer DynamoDB-Export-JSON

Die DynamoDB-Exporte mit dem AWS Glue Der DynamoDB-Exportkonnektor führt zu JSON-Dateien mit bestimmten verschachtelten Strukturen. Weitere Informationen finden Sie unter Datenobjekte. AWS Glue stellt eine DynamicFrame Transformation bereit, mit der solche Strukturen in ein easier-to-use Formular für nachgelagerte Anwendungen umgewandelt werden können.

Die Transformation kann auf zwei Arten aufgerufen werden. Beim Aufrufen einer Methode zum Lesen aus DynamoDB können Sie die Verbindungsoption `"dynamodb.simplifyDDBJson"` mit dem Wert `"true"` festlegen. Sie können die Transformation auch als eigenständige Methode aufrufen, die in der AWS Glue-Bibliothek verfügbar ist.

Betrachten Sie das folgende Schema, das durch einen DynamoDB-Export generiert wurde:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
```

```
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

Die Transformation `simplifyDDBJson` vereinfacht dies zu:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

> **Note**
>
> `simplifyDDBJson`ist in AWS Glue 3.0 und späteren Versionen verfügbar. Mit der Transformation `unnestDDBJson` können Sie auch das JSON aus einem DynamoDB-Export vereinfachen. Wir empfehlen Benutzern, von `unnestDDBJson` auf `simplifyDDBJson` umzusteigen.

Konfigurieren der Parallelisierung bei DynamoDB-Vorgängen

Um die Leistung zu verbessern, können Sie bestimmte für den DynamoDB-Konnektor verfügbare Parameter anpassen. Ihr Ziel bei der Optimierung der Parallelitätsparameter besteht darin, die Nutzung der bereitgestellten AWS Glue-Worker zu maximieren. Wenn Sie dann mehr Leistung benötigen, empfehlen wir Ihnen, Ihren Job zu skalieren, indem Sie die Anzahl der Aufgaben erhöhen. DPUs

Sie können die Parallelisierung in einem DynamoDB-Lesevorgang mit dem Parameter `dynamodb.splits` ändern, wenn Sie den ETL-Konnektor verwenden. Wenn Sie den Export-Konnektor zum Lesen verwenden, müssen Sie die Anzahl der Splits für die Spark-Executor-

Parallelisierung nicht konfigurieren. Sie können die Parallelisierung in einem DynamoDB-Schreibvorgang mit `dynamodb.output.numParallelTasks` ändern.

Lesen mit dem DynamoDB-ETL-Konnektor

Wir empfehlen Ihnen, die Berechnung von `dynamodb.splits` auf der Grundlage der in Ihrer Auftragskonfiguration festgelegten maximalen Anzahl von Workern und der folgenden `numSlots`-Berechnung durchzuführen. Bei einer automatischen Skalierung kann sich die tatsächliche Anzahl verfügbarer Worker unter dieser Obergrenze ändern. Weitere Informationen zur Festlegung der Höchstzahl an Workern finden Sie unter Anzahl der Worker (`NumberOfWorkers`) inthe section called “Konfiguration der Spark-Jobeigenschaften”.

- `numExecutors = NumberOfWorkers - 1`

  Für den Kontext ist ein Executor für den Spark-Treiber reserviert; andere Executors werden für die Datenverarbeitung verwendet.
- `numSlotsPerExecutor =`

  AWS Glue 3.0 and later versions
  - 4 wenn `WorkerType` gleich `G.1X`
  - 8 wenn `WorkerType` gleich `G.2X`
  - 16 wenn `WorkerType` gleich `G.4X`
  - 32 wenn `WorkerType` gleich `G.8X`

  AWS Glue 2.0 and legacy versions
  - 8 wenn `WorkerType` gleich `G.1X`
  - 16 wenn `WorkerType` gleich `G.2X`
- `numSlots = numSlotsPerExecutor * numExecutors`

Wir empfehlen Ihnen `dynamodb.splits` auf die Anzahl der verfügbaren Slots einzustellen, `numSlots`.

In DynamoDB schreiben

Der Parameter `dynamodb.output.numParallelTasks` wird verwendet, um die WCU pro Spark-Aufgabe anhand der folgenden Berechnung zu bestimmen:

```
permittedWcuPerTask = ( TableWCU * dynamodb.throughput.write.percent ) /
dynamodb.output.numParallelTasks
```

Der DynamoDB-Writer funktioniert am besten, wenn die Konfiguration die Anzahl der Spark-Aufgaben, die in DynamoDB schreiben, genau wiedergibt. In manchen Fällen müssen Sie die Standardberechnung außer Kraft setzen, um die Schreibleistung zu verbessern. Wenn Sie diesen Parameter nicht angeben, wird die zulässige WCU pro Spark-Aufgabe automatisch nach der folgenden Formel berechnet:

-
  - `numPartitions = dynamicframe.getNumPartitions()`
  - `numSlots` (wie zuvor in diesem Abschnitt definiert)
  - `numParallelTasks = min(numPartitions, numSlots)`
- Beispiel 1. DPU=10, =Standard. WorkerType Die Eingabe DynamicFrame hat 100 RDD-Partitionen.
  - `numPartitions = 100`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`
  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(100, 68) = 68`
- Beispiel 2. DPU = 10, =Standard. WorkerType Die Eingabe DynamicFrame hat 20 RDD-Partitionen.
  - `numPartitions = 20`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`
  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(20, 68) = 20`

> **Note**
>
> Jobs auf älteren AWS Glue-Versionen und solche, die Standard-Worker verwenden, erfordern unterschiedliche Methoden zur Berechnung der Anzahl der Slots. Wenn Sie die Leistung dieser Jobs optimieren müssen, empfehlen wir Ihnen, auf unterstützte AWS Glue-Versionen umzusteigen.

Referenz zur DynamoDB-Verbindungsoption

Bezeichnet eine Verbindung mit Amazon DynamoDB.

Die Anschlussmöglichkeiten bei einer Quellverbindung und eine Senkenverbindung unterscheiden sich.

„connectionType“: „dynamodb“ mit dem ETL-Konnektor als Quelle

Verwenden Sie die folgenden Verbindungsoptionen `"connectionType": "dynamodb"` als Quelle, wenn Sie den AWS Glue DynamoDB ETL-Konnektor verwenden:

- `"dynamodb.input.tableName"`: (Erforderlich) Die zu lesende DynamoDB-Tabelle.
- `"dynamodb.throughput.read.percent"` (optional): Der Prozentsatz der zu verwendenden Lesekapazitätseinheiten (Read Capacity Units, RCU). Der Standard ist auf „0,5“ gesetzt. Zulässigen Werte sind „0,1“ bis „1,5“ inklusive.
  - `0.5`steht für die Standard-Leserate, d. h. AWS Glue versucht, die Hälfte der Lesekapazität der Tabelle zu verbrauchen. Wenn Sie den obigen Wert erhöhen`0.5`, AWS Glue erhöht die Anforderungsrate. Wenn Sie den Wert unter diesen Wert verringern, `0.5` verringert sich die Leseanforderungsrate. (Die tatsächliche Leserate variiert abhängig von Faktoren wie der Tatsache, ob eine einheitliche Verteilung des Schlüssels in der DynamoDB-Tabelle vorliegt.)
  - Wenn sich die DynamoDB-Tabelle im On-Demand-Modus befindet, AWS Glue behandelt die Lesekapazität der Tabelle als 40000. Zum Exportieren einer großen Tabelle empfehlen wir, Ihre DynamoDB-Tabelle in den On-Demand-Modus zu ändern.
- `"dynamodb.splits"`: (Optional) Definiert die Anzahl der Teile, in die diese DynamoDB-Tabelle beim Lesen partitioniert wird. Der Standard ist auf „1“ gesetzt. Zulässigen Werte sind „1“ bis „1,000,000“ inklusive.

  1 stellt dar, dass es keine Parallelität gibt. Es wird dringend empfohlen, dass Sie einen größeren Wert für eine bessere Leistung angeben, indem Sie die folgende Formel verwenden. Weitere Informationen zur korrekten Einstellung eines Werts finden Sie unter the section called “Parallelisierung in DynamoDB”.
- `"dynamodb.sts.roleArn"`: (Optional) Die IAM-Rolle ARN, die für den kontoübergreifenden Zugriff angenommen werden soll. Dieser Parameter ist verfügbar in AWS Glue 1.0 oder höher.
- `"dynamodb.sts.roleSessionName"`: (Optional) Name der STS-Sitzung. Die Standardeinstellung ist "glue-dynamodb-read-sts-session“. Dieser Parameter ist verfügbar in AWS Glue 1.0 oder höher.

„connectionType“: „dynamodb“ mit dem AWS Glue DynamoDB-Exportkonnektor als Quelle

Verwenden Sie die folgenden Verbindungsoptionen mit „connectionType“: „dynamodb“ als Quelle, wenn Sie AWS Glue DynamoDB-Exportconnector, der nur verfügbar ist für AWS Glue ab Version 2.0:

- `"dynamodb.export"`: (Erforderlich) Ein Zeichenfolgenwert:
  - Wenn auf gesetzt, `ddb` aktiviert AWS Glue DynamoDB-Exportconnector, bei dem ein neuer aufgerufen `ExportTableToPointInTimeRequest` wird während des AWS Glue Arbeit. Ein neuer Export wird mit dem Speicherort generiert, der von `dynamodb.s3.bucket` an `dynamodb.s3.prefix` übergeben wurde.
  - Wenn auf gesetzt, `s3` aktiviert es AWS Glue DynamoDB-Exportconnector überspringt jedoch die Erstellung eines neuen DynamoDB-Exports und verwendet stattdessen `dynamodb.s3.bucket` und `dynamodb.s3.prefix` als Amazon S3 S3-Speicherort für einen früheren Export dieser Tabelle.
- `"dynamodb.tableArn"`: (Erforderlich) Die zu lesende DynamoDB-Tabelle.
- `"dynamodb.unnestDDBJson"`: (Optional) Standard: falsch. Zulässige Werte: Boolesch. Wenn der Wert auf „true“ festgelegt ist, wird die Verschachtelung der DynamoDB-JSON-Struktur, die in den Exporten vorhanden ist, durch eine Transformation aufgehoben. Es ist ein Fehler, `"dynamodb.unnestDDBJson"` und `"dynamodb.simplifyDDBJson"` gleichzeitig auf „true“ zu setzen. In AWS Glue 3.0 und späteren Versionen empfehlen wir die Verwendung`"dynamodb.simplifyDDBJson"`, um das Verhalten bei der Vereinfachung von DynamoDB-Zuordnungstypen zu verbessern. Weitere Informationen finden Sie unter the section called “Vereinfachung der Verwendung einer DynamoDB-Export-JSON”.
- `"dynamodb.simplifyDDBJson"`: (Optional) Standard: falsch. Zulässige Werte: Boolesch. Wenn der Wert auf „true“ festgelegt ist, wird eine Transformation zur Vereinfachung der DynamoDB-JSON-Struktur, die in den Exporten vorhanden ist, durchgeführt. Dies hat den gleichen Zweck wie die Option `"dynamodb.unnestDDBJson"`, bietet jedoch eine bessere Unterstützung für DynamoDB-Zuordnungstypen oder sogar verschachtelte Zuordnungstypen in Ihrer DynamoDB-Tabelle. Diese Option ist in AWS Glue 3.0 und späteren Versionen verfügbar. Es ist ein Fehler, `"dynamodb.unnestDDBJson"` und `"dynamodb.simplifyDDBJson"` gleichzeitig auf „true“ zu setzen. Weitere Informationen finden Sie unter the section called “Vereinfachung der Verwendung einer DynamoDB-Export-JSON”.
- `"dynamodb.s3.bucket"`: (Optional) Gibt den Speicherort des Amazon S3 Bucket an, in dem der DynamoDB `ExportTableToPointInTime`-Prozess durchgeführt werden soll. Das Dateiformat für den Export ist DynamoDB JSON.

  - `"dynamodb.s3.prefix"`: (Optional) Gibt den Speicherort des Amazon S3-Präfixes im AmazonS3 Bucket an, in dem die DynamoDB `ExportTableToPointInTime`-Lasten gespeichert werden sollen. Wenn `dynamodb.s3.prefix` weder angegeben noch angegeben `dynamodb.s3.bucket` werden, werden diese Werte standardmäßig auf den Speicherort des temporären Verzeichnisses gesetzt, der in AWS Glue Auftragskonfiguration. Weitere Informationen finden Sie unter Von AWS Glue verwendete Spezialparameter.
  - `"dynamodb.s3.bucketOwner"`: Zeigt den Bucket-Besitzer an, der für den kontoübergreifenden Amazon-S3-Zugriff benötigt wird.
- `"dynamodb.sts.roleArn"`: (Optional) Der IAM-Rollen-ARN, der für den kontoübergreifenden Zugriff und/oder den regionsübergreifenden Zugriff auf die DynamoDB-Tabelle übernommen werden soll. Hinweis: Dieselbe IAM-Rolle ARN wird für den Zugriff auf den für die `ExportTableToPointInTime`-Anfrage angegebenen Amazon S3-Speicherort verwendet.
- `"dynamodb.sts.roleSessionName"`: (Optional) Name der STS-Sitzung. Die Standardeinstellung ist "glue-dynamodb-read-sts-session“.
- `"dynamodb.exportTime"`: (Optional) Gültige Werte: Zeichenfolgen, die ISO-8601-Instanzen darstellen. A, point-in-time an dem der Export erfolgen soll.
- `"dynamodb.sts.region"`: (Erforderlich, wenn ein regionsübergreifender Aufruf über einen regionalen Endpunkt getätigt wird). Die Region, in der sich die DynamoDB-Tabelle befindet, die Sie lesen möchten.

„connectionType“: „dynamodb“ mit dem ETL-Konnektor als Senke

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "dynamodb"` als Senke:

- `"dynamodb.output.tableName"`: (Erforderlich) Die DynamoDB-Tabelle, in die geschrieben werden soll.
- `"dynamodb.throughput.write.percent"` (optional): Der Prozentsatz der zu verwendenden Schreibkapazitätseinheiten (Write Capacity Units, WCU). Der Standard ist auf „0,5“ gesetzt. Zulässigen Werte sind „0,1“ bis „1,5“ inklusive.
  - `0.5`steht für die Standard-Schreibrate, was bedeutet, dass AWS Glue versucht, die Hälfte der Schreibkapazität der Tabelle zu verbrauchen. Wenn Sie den Wert über 0,5 erhöhen, AWS Glue erhöht die Anforderungsrate; wenn der Wert unter 0,5 sinkt, sinkt die Schreibanforderungsrate. (Die tatsächliche Schreibrate variiert abhängig von Faktoren wie der Tatsache, ob eine einheitliche Verteilung des Schlüssels in der DynamoDB-Tabelle vorliegt.)

  - Wenn sich die DynamoDB-Tabelle im On-Demand-Modus befindet, AWS Glue behandelt die Schreibkapazität der Tabelle als. `40000` Zum Importieren einer großen Tabelle empfehlen wir, Ihre DynamoDB-Tabelle in den On-Demand-Modus zu ändern.
- `"dynamodb.output.numParallelTasks"`: (Optional) definiert, wie viele parallele Aufgaben gleichzeitig in DynamoDB geschrieben werden. Wird verwendet, um berechtigende WCU pro Spark-Aufgabe zu berechnen. In den meisten Fällen berechnet AWS Glue einen angemessenen Standardwert für diesen Wert. Weitere Informationen finden Sie unter the section called “Parallelisierung in DynamoDB”.
- `"dynamodb.output.retry"`: (Optional) Definiert die Anzahl erneuter Versuche, die durchgeführt werden, wenn es eine `ProvisionedThroughputExceededException` aus DynamoDB gibt. Der Standard ist auf „10“ gesetzt.
- `"dynamodb.sts.roleArn"`: (Optional) Die IAM-Rolle ARN, die für den kontoübergreifenden Zugriff angenommen werden soll.
- `"dynamodb.sts.roleSessionName"`: (Optional) Name der STS-Sitzung. Die Standardeinstellung ist auf "glue-dynamodb-write-sts-session“ gesetzt.

Konten- und regionenübergreifender Zugriff auf DynamoDB-Tabellen

AWS Glue-ETL-Aufträge unterstützen sowohl Konten- und regionenübergreifenden Zugriff auf DynamoDB-Tabellen. AWS Glue ETL-Jobs unterstützen sowohl das Lesen von Daten aus der DynamoDB-Tabelle eines anderen AWS Kontos als auch das Schreiben von Daten in die DynamoDB-Tabelle eines anderen AWS Kontos. AWS Glueunterstützt auch das Lesen aus einer DynamoDB-Tabelle in einer anderen Region und das Schreiben in eine DynamoDB-Tabelle in einer anderen Region. Dieser Abschnitt enthält Anweisungen zum Einrichten des Zugriffs und enthält ein Beispielskript.

Die Verfahren in diesem Abschnitt verweisen auf ein IAM-Lernprogramm zum Erstellen einer IAM-Rolle und zum Erteilen des Zugriffs auf die Rolle. Das Tutorial bespricht auch die Übernahme einer Rolle, aber hier werden Sie stattdessen ein Auftragsskript verwenden, um die Rolle in AWS Glue anzunehmen. Dieses Tutorial enthält auch Informationen über allgemeine kontoübergreifende Praktiken. Weitere Informationen finden Sie unter Tutorial: AWS Kontoübergreifendes Delegieren des Zugriffs mithilfe von IAM-Rollen im IAM-Benutzerhandbuch.

Erstellen einer Rolle

Folgen Sie Schritt 1 des Tutorials, um eine IAM-Rolle in Konto A zu erstellen. Wenn Sie die Berechtigungen der Rolle definieren, können Sie festlegen, ob Sie bestehende Richtlinien anhängen

möchten, z. B. DynamoDB`AmazonDynamoDBReadOnlyAccess`, oder ob die Rolle read/write DynamoDB `AmazonDynamoDBFullAccess` zugewiesen werden soll. Im folgenden Beispiel sehen Sie das Erstellen einer Rolle namens`DynamoDBCrossAccessRole`, mit der Berechtigungsrichtlinie `AmazonDynamoDBFullAccess`.

Erteilen der Zugriffsberechtigung auf die Rolle

Befolgen Sie Schritt 2 im Tutorial im IAM-Benutzerhandbuch, damit Konto B zur neu erstellten Rolle wechseln kann. Im folgenden Beispiel wird eine neue Richtlinie mit der folgenden Anweisung erstellt:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "<DynamoDBCrossAccessRole's ARN>"
    }
}
```

Anschließend können Sie diese Richtlinie an die Richtlinie anhängen, die group/role/user Sie für den Zugriff auf DynamoDB verwenden möchten.

Übernehmen Sie die Rolle im AWS Glue-Auftragsskript

Jetzt können Sie sich bei Konto B anmelden und einen AWS Glue-Auftrag erstellen. Informationen zum Erstellen eines Auftrags finden Sie in den Anweisungen unter Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue.

Im Job-Skript müssen Sie die `dynamodb.sts.roleArn`-Parameter verwenden, um die `DynamoDBCrossAccessRole`-Rolle anzunehmen. Wenn Sie diese Rolle annehmen, können Sie die temporären Anmeldeinformationen für den Zugriff auf DynamoDB in Konto B erhalten. Sehen Sie sich die Beispielskripts an.

Für kontenübergreifendes Lesen in verschiedenen Regionen (ETL Connector):

```
import sys
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
    "dynamodb.region": "us-east-1",
    "dynamodb.input.tableName": "test_source",
    "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
dyf.show()
job.commit()
```

Für kontenübergreifendes Lesen in verschiedenen Regionen (ELT Connector):

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": "<test_source ARN>",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
dyf.show()
job.commit()
```

Für kontenübergreifendes Lesen und Schreiben in verschiedenen Regionen:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-east-1",
        "dynamodb.input.tableName": "test_source"
    }
)
dyf.show()

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-west-2",
        "dynamodb.output.tableName": "test_sink",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)

job.commit()
```

Kinesis-Verbindungen

Sie können eine Kinesis-Verbindung zum Lesen und Schreiben in Amazon Kinesis Kinesis-Datenstreams verwenden, indem Sie Informationen verwenden, die in einer Datenkatalogtabelle gespeichert sind, oder indem Sie Informationen bereitstellen, um direkt auf den Datenstream zuzugreifen. Sie können Informationen aus Kinesis in einen Spark einlesen DataFrame und sie dann in einen AWS Glue DynamicFrame umwandeln. Sie können in einem JSON-Format in Kinesis

schreiben DynamicFrames . Wenn Sie direkt auf den Datenstrom zugreifen, verwenden Sie diese Optionen, um Informationen zum Zugriff auf den Datenstrom bereitzustellen.

Wenn Sie `getCatalogSource` oder `create_data_frame_from_catalog` verwenden, um Einträge aus einer Kinesis-Streamingquelle zu verbrauchen, enthält der Auftrag die Informationen zu Data-Catalog-Datenbank und Tabellennamen und kann diese verwenden, um einige grundlegende Parameter für das Lesen aus der Kinesis-Streaming-Quelle zu erhalten. Wenn Sie `getSource`, `getSourceWithFormat`, `createDataFrameFromOptions` oder `create_data_frame_from_options` verwenden, müssen Sie diese grundlegenden Parameter mithilfe der hier beschriebenen Verbindungsoptionen angeben.

Sie können die Verbindungsoptionen für Kinesis mit den folgenden Argumenten für die angegebenen Methoden in der `GlueContext`-Klasse angeben.

- Scala
  - `connectionOptions`: mit `getSource`, `createDataFrameFromOptions`, `getSink` verwenden
  - `additionalOptions`: mit `getCatalogSource`, `getCatalogSink` verwenden
  - `options`: mit `getSourceWithFormat`, `getSinkWithFormat` verwenden
- Python
  - `connection_options`: mit `create_data_frame_from_options`, `write_dynamic_frame_from_options` verwenden
  - `additional_options`: mit `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog` verwenden
  - `options`: mit `getSource`, `getSink` verwenden

Hinweise und Einschränkungen zu Streaming-ETL-Aufträgen finden Sie unter the section called “Hinweise zu und Einschränkungen für Streaming-ETL”.

Kinesis konfigurieren

Um in einem AWS Glue Spark-Job eine Verbindung zu einem Kinesis-Datenstream herzustellen, benötigen Sie einige Voraussetzungen:

- Beim Lesen muss der AWS Glue-Job über IAM-Berechtigungen auf Lesezugriffsebene für den Kinesis-Datenstream verfügen.

- Beim Schreiben muss der AWS Glue-Job über IAM-Berechtigungen auf Schreibzugriffsebene für den Kinesis-Datenstream verfügen.

In bestimmten Fällen müssen Sie zusätzliche Voraussetzungen konfigurieren:

- Wenn Ihr AWS Glue-Job mit zusätzlichen Netzwerkverbindungen konfiguriert ist (in der Regel, um eine Verbindung zu anderen Datensätzen herzustellen) und eine dieser Verbindungen Amazon VPC-Netzwerkoptionen bietet, leitet dies Ihren Job an, über Amazon VPC zu kommunizieren. In diesem Fall müssen Sie auch Ihren Kinesis-Datenstrom für die Kommunikation über Amazon VPC konfigurieren. Sie können dies tun, indem Sie einen Schnittstellen-VPC-Endpunkt zwischen Ihrer Amazon VPC und dem Kinesis-Datenstrom erstellen. Weitere Informationen finden Sie unter Verwenden von Kinesis Data Streams mit Schnittstellen-VPC-Endpunkten.
- Wenn Sie Amazon Kinesis Data Streams in einem anderen Konto angeben, müssen Sie die Rollen und Richtlinien einrichten, um den kontoübergreifenden Zugriff zu ermöglichen. Weitere Informationen finden Sie unter Beispiel: Aus einem Kinesis Stream in einem anderen Konto lesen.

Weitere Informationen zu den Voraussetzungen für Streaming-ETL-Aufträgen finden Sie unter the section called “Streaming-ETL-Aufträge”.

Beispiel: Lesen aus Kinesis-Streams

Beispiel: Lesen aus Kinesis-Streams

Verwendet in Verbindung mit the section called “forEachBatch”.

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Beispiel: In Kinesis-Streams schreiben

Beispiel: Lesen aus Kinesis-Streams

Verwendet in Verbindung mit the section called “forEachBatch”.

Beispiel für Amazon-Kinesis-Streaming-Quelle:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Referenz zur Kinesis-Verbindungsoption

Bezeichnet Verbindungoptionen zu Amazon Kinesis Data Streams.

Verwenden Sie die folgenden Verbindungsoptionen für Kinesis-Streamingdatenquellen:

- `"streamARN"` (Erforderlich) Wird zum Lesen/Schreiben verwendet. Der ARN des Kinesis-Datenstroms.
- `"classification"` (Zum Lesen erforderlich) Wird zum Lesen verwendet. Das von den Daten im Datensatz verwendete Dateiformat. Erforderlich, sofern nicht in Data Catalog angegeben.
- `"streamName"` – (Optional) Wird zum Lesen verwendet. Der Name eines Kinesis-Datenstroms, aus dem gelesen wird. Wird mit `endpointUrl` verwendet.
- `"endpointUrl"` – (Optional) Wird zum Lesen verwendet. Standard: "“. https://kinesis.us-east-1.amazonaws.com Der AWS Endpunkt des Kinesis-Streams. Sie müssen dies nicht ändern, es sei denn, Sie stellen eine Verbindung zu einer bestimmten Region her.
- `"partitionKey"` – (Optional) Wird zum Schreiben verwendet. Der Kinesis-Partitionsschlüssel, der bei der Erstellung von Datensätzen verwendet wird.
- `"delimiter"` (Optional) Wird zum Lesen verwendet. Das verwendete Werttrennzeichen, wenn `classification` CSV ist. Der Standardwert ist „,„.
- `"startingPosition"`: (Optional) Wird zum Lesen verwendet. Die Ausgangsposition im Kinesis Data Stream, von dem Daten gelesen werden sollen. Die möglichen Werte sind `"latest"`,

`"trim_horizon"`, `"earliest"` oder eine Zeitstempelzeichenfolge im UTC-Format im Muster `yyyy-mm-ddTHH:MM:SSZ` (wobei `Z` einen UTC-Zeitzonenversatz mit einem +/- darstellt. Zum Beispiel „2023-04-04T08:00:00-04:00“). Der Standardwert ist `"latest"`. Hinweis: Die Timestamp-Zeichenfolge im UTC-Format für `"startingPosition"` wird nur für AWS Glue Version 4.0 oder höher unterstützt.

- `"failOnDataLoss"`: (Optional) Lassen Sie den Auftrag fehlschlagen, wenn ein aktiver Shard fehlt oder abgelaufen ist. Der Standardwert ist `"false"`.
- `"awsSTSRoleARN"`: (Optional) Wird zum Schreiben/Lesen verwendet. Der Amazon-Ressourcenname (ARN) der Rolle, die mithilfe von AWS Security Token Service (AWS STS) übernommen werden soll. Diese Rolle muss über Berechtigungen zum Beschreiben oder Lesen von Datensatzoperationen für den Kinesis-Datenstrom verfügen. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit `"awsSTSSessionName"`.
- `"awsSTSSessionName"`: (Optional) Wird zum Schreiben/Lesen verwendet. Ein Bezeichner für die Sitzung, die die Rolle unter Verwendung von AWS STSübernimmt. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit `"awsSTSRoleARN"`.
- `"awsSTSEndpoint"`: (Optional) Der AWS STS Endpunkt, der verwendet werden soll, wenn eine Verbindung zu Kinesis mit einer angenommenen Rolle hergestellt wird. Dies ermöglicht die Verwendung des regionalen AWS STS Endpunkts in einer VPC, was mit dem globalen Standardendpunkt nicht möglich ist.
- `"maxFetchTimeInMs"`: (Optional) Wird zum Lesen verwendet. Die maximale Zeit, die der Job Executor benötigt, um Datensätze für den aktuellen Batch aus dem Kinesis-Datenstream zu lesen, angegeben in Millisekunden (ms). Innerhalb dieser Zeit können mehrere `GetRecords` API-Aufrufe getätigt werden. Der Standardwert ist `1000`.
- `"maxFetchRecordsPerShard"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl von Datensätzen, die pro Shard im Kinesis-Datenstrom pro Mikrobatch abgerufen werden können. Hinweis: Der Client kann dieses Limit überschreiten, wenn der Streaming-Job bereits zusätzliche Datensätze von Kinesis gelesen hat (im selben Get-Records-Aufruf). Wenn es streng sein `maxFetchRecordsPerShard` muss, muss es ein Vielfaches von sein. `maxRecordPerRead` Der Standardwert ist `100000`.
- `"maxRecordPerRead"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl von Datensätzen, die bei jeder `getRecords`-Operation aus dem Kinesis-Datenstrom abgerufen werden sollen. Der Standardwert ist `10000`.

- `"addIdleTimeBetweenReads"`: (Optional) Wird zum Lesen verwendet. Fügt eine Zeitverzögerung zwischen zwei aufeinanderfolgenden `getRecords`-Operationen ein. Der Standardwert ist `"False"`. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.
- `"idleTimeBetweenReadsInMs"`: (Optional) Wird zum Lesen verwendet. Die minimale Zeitverzögerung zwischen zwei aufeinanderfolgenden `getRecords`-Operationen, angegeben in Millisekunden (ms). Der Standardwert ist `1000`. Diese Option ist nur für Glue 2.0 und höher konfigurierbar.
- `"describeShardInterval"`: (Optional) Wird zum Lesen verwendet. Das minimale Zeitintervall zwischen zwei `ListShards`-API-Aufrufen für die Erwägung eines erneuten Shardings durch Ihr Skript. Weitere Informationen finden Sie unter Strategies for Resharding (Strategien für das Resharding) im Entwicklerhandbuch für Amazon Kinesis Data Streams. Der Standardwert ist `1s`.
- `"numRetries"`: (Optional) Wird zum Lesen verwendet. Die maximale Anzahl erneuter Versuche für API-Aufrufe von Kinesis Data Streams. Der Standardwert ist `3`.
- `"retryIntervalMs"`: (Optional) Wird zum Lesen verwendet. Die Abkühlzeit (angegeben in ms) vor dem erneuten Versuch des API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `1000`.
- `"maxRetryIntervalMs"`: (Optional) Wird zum Lesen verwendet. Die maximale Abkühlzeit (angegeben in ms) zwischen zwei wiederholten Versuchen eines API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `10000`.
- `"avoidEmptyBatches"`: (Optional) Wird zum Lesen verwendet. Vermeidet das Erstellen eines leeren Mikrobatchauftrags, indem vor dem Start des Batches im Kinesis Data Stream nach ungelesenen Daten gesucht wird. Der Standardwert ist `"False"`.
- `"schema"`: (Erforderlich, wenn inferSchema auf „false“ festgelegt ist) Wird zum Lesen verwendet. Das zu verwendende Schema für die Verarbeitung der Nutzlast. Wenn die Klassifizierung `avro` ist, muss das bereitgestellte Schema im Avro-Schemaformat vorliegen. Wenn die Klassifizierung nicht `avro` ist, muss das bereitgestellte Schema im DDL-Schemaformat vorliegen.

  Im Folgenden finden Sie Beispiele für Schemata.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
  ```

```
  {
    "type":"record",
    "name":"test",
    "fields":
    [
      {
        "name":"_id",
        "type":"string"
      },
      {
        "name":"index",
        "type":
        [
          "int",
          "string",
          "float"
        ]
      }
    ]
  }
}
```

- `"inferSchema"`: (Optional) Wird zum Lesen verwendet. Der Standardwert von "false". Wenn auf „true“ gesetzt, wird das Schema zur Laufzeit von der Nutzlast in `foreachbatch` erkannt.
- `"avroSchema"`: (Veraltet) Wird zum Lesen verwendet. Der Parameter, der verwendet wird, um ein Schema von Avro-Daten anzugeben, wenn das Avro-Format verwendet wird. Dieser Parameter ist jetzt veraltet. Verwenden Sie den Parameter `schema`.
- `"addRecordTimestamp"`: (Optional) Wird zum Lesen verwendet. Wenn diese Option auf 'true' gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die die Uhrzeit angibt, zu der der entsprechende Datensatz mit dem Stream empfangen wurde. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"emitConsumerLagMetrics"`: (Optional) Wird zum Lesen verwendet. Wenn die Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, der vom Stream empfangen wurde, und dem Zeitpunkt, zu dem er eingeht, ausgegeben AWS Glue zu CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"fanoutConsumerARN"`: (Optional) Wird zum Lesen verwendet. Der ARN eines Kinesis-Stream-Verbrauchers für den in `streamARN` angegebenen Stream. Wird verwendet, um den erweiterten

Fan-Out-Modus für Ihre Kinesis-Verbindung zu aktivieren. Weitere Informationen zur Nutzung eines Kinesis-Streams mit erweitertem Fan-Out finden Sie unter the section called “Verwendung von erweitertem Fan-Out in Kinesis-Streaming-Aufträgen”.

- `"recordMaxBufferedTime"` – (Optional) Wird zum Schreiben verwendet. Standard: 1.000 (ms). Maximale Dauer, für die ein Datensatz gepuffert wird, während er darauf wartet, geschrieben zu werden.
- `"aggregationEnabled"` – (Optional) Wird zum Schreiben verwendet. Standard: true Gibt an, ob Datensätze aggregiert werden sollen, bevor sie an Kinesis gesendet werden.
- `"aggregationMaxSize"` – (Optional) Wird zum Schreiben verwendet. Standard: 51.200 (Byte). Wenn ein Datensatz dieses Limit übersteigt, wird der Aggregator umgangen. Hinweis Kinesis erzwingt ein Limit von 50 KB für die Größe eines Datensatzes. Wenn Sie diesen Wert auf mehr als 50 KB festlegen, werden übergroße Datensätze von Kinesis zurückgewiesen.
- `"aggregationMaxCount"` – (Optional) Wird zum Schreiben verwendet. Standard: 4.294.967.295. Maximale Anzahl von Elementen, die in einen aggregierten Datensatz gepackt werden sollen.
- `"producerRateLimit"` – (Optional) Wird zum Schreiben verwendet. Standard: 150 (%). Beschränkt den Durchsatz pro Shard, der von einem einzelnen Produzenten (z. B. Ihrem Auftrag) gesendet wird, als prozentualen Wert des Backend-Limits.
- `"collectionMaxCount"` – (Optional) Wird zum Schreiben verwendet. Standard: 500. Maximale Anzahl von Elementen, die in eine PutRecords Anfrage gepackt werden sollen.
- `"collectionMaxSize"` – (Optional) Wird zum Schreiben verwendet. Standard: 5.242.880 (Byte). Maximale Datenmenge, die mit einer PutRecords Anfrage gesendet werden kann.

Verwendung von erweitertem Fan-Out in Kinesis-Streaming-Aufträgen

Ein erweiterter Fan-Out-Verbraucher kann Datensätze von einem Kinesis-Stream mit einem dedizierten Durchsatz empfangen, der größer sein kann als bei typischen Anwendern. Dies erfolgt durch die Optimierung des Übertragungsprotokolls, das zur Bereitstellung von Daten an einen Kinesis-Verbraucher, beispielsweise Ihren Auftrag, verwendet wird. Weitere Informationen zu Kinesis Enhanced Fan-Out finden Sie in der Kinesis-Dokumentation.

Im erweiterten Fan-Out-Modus gelten die `maxRecordPerRead`- und `idleTimeBetweenReadsInMs`-Verbindungsoptionen nicht mehr, da diese Parameter bei Verwendung des erweiterten Fan-Outs nicht konfigurierbar sind. Die Konfigurationsoptionen für Wiederholungsversuche funktionieren wie beschrieben.

Verwenden Sie die folgenden Verfahren, um das erweiterte Fanout für Ihren Streaming-Auftrag zu aktivieren und zu deaktivieren. Sie sollten für jeden Auftrag, der Daten aus Ihrem Stream konsumiert, einen Stream-Anwender registrieren.

So aktivieren Sie einen verbesserten Fan-Out-Verbrauch für Ihren Auftrag:

1. Registrieren Sie mithilfe der Kinesis-API einen Stream-Verbraucher für Ihren Auftrag. Befolgen Sie die Anweisungen zum Registrieren eines Verbrauchers mit erweitertem Fan-Out mithilfe der Kinesis-Data-Streams-API in der Kinesis-Dokumentation. Sie müssen nur dem ersten Schritt folgen - dem Anrufen RegisterStreamConsumer. Ihre Anfrage sollte einen ARN zurückgeben,*consumerARN*.
2. Stellen Sie die Verbindungsoption *consumerARN* in `fanoutConsumerARN` Ihren Argumenten für die Verbindungsmethode auf ein.
3. Starten Sie Ihren Auftrag neu.

So deaktivieren Sie den erweiterten Fanout-Verbrauch für Ihren Auftrag:

1. Entfernen Sie die `fanoutConsumerARN`-Verbindungsoption aus Ihrem Methodenaufruf.
2. Starten Sie Ihren Auftrag neu.
3. Befolgen Sie die Anweisungen zum Abmelden eines Verbrauchers in der Kinesis-Dokumentation. Diese Anweisungen gelten für die Konsole, können aber auch über die Kinesis-API erreicht werden. Weitere Informationen zur Abmeldung von Stream-Consumer über die Kinesis-API finden Sie DeregisterStreamConsumerin der Kinesis-Dokumentation.

Amazon-S3-Verbindungen

Sie können AWS Glue for Spark verwenden, um Dateien in Amazon S3 zu lesen und zu schreiben. AWS Glue for Spark unterstützt viele gängige Datenformate, die standardmäßig in Amazon S3 gespeichert sind, darunter CSV, Avro, JSON, Orc und Parquet. Weitere Informationen zu unterstützten Datenformaten finden Sie unter the section called “Pfad-Formatoptionen”. Jedes Datenformat unterstützt möglicherweise einen anderen Satz von AWS Glue-Funktionen. Informieren Sie sich auf der Seite für Ihr Datenformat über die Besonderheiten der Feature-Unterstützung. Darüber hinaus können Sie versionierte Dateien lesen und schreiben, die in den Data-Lake-Frameworks Hudi, Iceberg und Delta Lake gespeichert sind. Weitere Informationen zu Data Lake-Frameworks finden Sie unter the section called “Data-Lake-Frameworks”.

Mit AWS Glue können Sie Ihre Amazon S3 S3-Objekte beim Schreiben in eine Ordnerstruktur partitionieren und sie dann partitionsweise abrufen, um die Leistung mithilfe einer einfachen Konfiguration zu verbessern. Sie können die Konfiguration auch so festlegen, dass kleine Dateien beim Transformieren Ihrer Daten gruppiert werden, um die Leistung zu verbessern. Sie können in Amazon S3 lesen, `bzip2` schreiben und `gzip` archivieren.

Themen

- Konfiguration von S3-Verbindungen
- Referenz zur Amazon-S3-Verbindungsoption
- Veraltete Verbindungssyntaxen für Datenformate
- Ausschließen von Amazon-S3-Speicherklassen
- Partitionen für die ETL-Ausgabe verwalten in AWS Glue
- Zusammenfassen von Eingabedateien in größeren Gruppen beim Lesen
- Amazon-VPC-Endpunkte für Amazon S3

Konfiguration von S3-Verbindungen

Um in einem AWS Glue with Spark-Job eine Verbindung zu Amazon S3 herzustellen, benötigen Sie einige Voraussetzungen:

- Der AWS Glue-Job muss über IAM-Berechtigungen für relevante Amazon S3 S3-Buckets verfügen.

In bestimmten Fällen müssen Sie zusätzliche Voraussetzungen konfigurieren:

- Bei der Konfiguration des kontoübergreifenden Zugriffs sind entsprechende Zugriffskontrollen für den Amazon-S3-Bucket erforderlich.
- Aus Sicherheitsgründen können Sie Ihre Amazon-S3-Anfragen über eine Amazon VPC weiterleiten. Dieser Ansatz kann zu Herausforderungen in Bezug auf Bandbreite und Verfügbarkeit führen. Weitere Informationen finden Sie unter the section called “Amazon-VPC-Endpunkte für Amazon S3”.

Referenz zur Amazon-S3-Verbindungsoption

Bezeichnet eine Verbindung mit Amazon S3.

Da Amazon S3 Dateien und nicht Tabellen verwaltet, müssen Sie zusätzlich zur Angabe der in diesem Dokument bereitgestellten Verbindungseigenschaften auch zusätzliche Konfigurationen für Ihren Dateityp angeben. Sie geben diese Informationen über Datenformatoptionen an. Weitere Informationen zu den Formatoptionen finden Sie unter the section called “Pfad-Formatoptionen”. Sie können diese Informationen auch durch die Integration mit dem AWS Glue Data Catalog angeben.

Betrachten Sie als Beispiel für die Unterscheidung zwischen Verbindungsoptionen und Formatoptionen die Art und Weise, wie die the section called “create_dynamic_frame_from_options”-Methode `connection_type`, `connection_options`, `format` und `format_options` verwendet. In diesem Abschnitt werden speziell die für `connection_options` bereitgestellten Parameter erläutert.

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "s3"`:

- `"paths"`: (Erforderlich) Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- `"exclusions"`: (Optional) Eine Zeichenfolge, die eine JSON-Liste der auszuschließenden Glob-Muster im Unix-Stil enthält. Beispiel: `"[\"**.pdf\"]"` schließt alle PDF-Dateien aus. Weitere Informationen zur Glob-Syntax finden Sie unter AWS Glue unterstützt, siehe Muster einschließen und ausschließen.
- `"compressionType"` oder `"compression"`: (Optional) Gibt an, wie die Daten komprimiert werden. Verwenden von `"compressionType"` für Amazon-S3-Quellen und `"compression"` für Amazon-S3-Ziele. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip2"`). Für bestimmte Formate werden möglicherweise zusätzliche Komprimierungsformate unterstützt. Einzelheiten zur Feature-Unterstützung finden Sie auf der Seite zum Datenformat.
- `"groupFiles"`: (Optional) Die Gruppierung von Dateien ist standardmäßig aktiviert, wenn die Eingabe mehr als 50 000 Dateien enthält. Um die Gruppierung mit weniger als 50 000 Dateien zu aktivieren, setzen Sie diesen Parameter auf `"inPartition"`. Um die Gruppierung zu deaktivieren, wenn mehr als 50 000 Dateien vorhanden sind, setzen Sie diesen Parameter auf `"none"`.
- `"groupSize"`: (Optional) Die Größe der Zielgruppe in Bytes. Der Standardwert wird anhand der Größe der Eingabedaten und der Größe des Clusters berechnet. Wenn es weniger als 50 000 Eingabedateien gibt, muss `"groupFiles"` auf `"inPartition"` gesetzt werden, damit dies wirksam wird.
- `"recurse"`: (Optional) Wenn auf true gesetzt, werden Dateien in allen Unterverzeichnissen unter den angegebenen Pfaden rekursiv gelesen.

- `"maxBand"`: (Optional, erweitert) – Diese Option steuert die Dauer in Millisekunden, nach der die `s3`-Auflistung wahrscheinlich konsistent ist. Dateien mit Änderungszeitstempeln, die innerhalb der letzten `maxBand` Millisekunden liegen, werden vor allem bei der Verwendung von `JobBookmarks` zur Berücksichtigung der letztendlichen Konsistenz von Amazon S3 verfolgt. Die meisten Benutzer müssen diese Option nicht festlegen. Der Standardwert ist 900 000 Millisekunden oder 15 Minuten
- `"maxFilesInBand"`: (Optional, erweitert) – Diese Option gibt die maximale Anzahl von Dateien an, die aus den letzten `maxBand` Sekunden gespeichert werden sollen. Wird diese Anzahl überschritten, werden zusätzliche Dateien übersprungen und erst bei der nächsten Auftragsausführung verarbeitet. Die meisten Benutzer müssen diese Option nicht festlegen.
- `"isFailFast"`: (Optional) Diese Option bestimmt, ob ein AWS Glue Der ETL-Job löst Ausnahmen beim Analysieren von Lesern aus. Wenn `true`, schlagen Aufträge schnell fehl, wenn vier Wiederholungen der Spark-Aufgabe die Daten nicht korrekt parsen.
- `"catalogPartitionPredicate"`: (Optional) Wird zum Lesen verwendet. Der Inhalt einer SQL-`WHERE`-Klausel. Wird beim Lesen aus Data-Catalog-Tabellen mit einer sehr großen Anzahl von Partitionen verwendet. Ruft passende Partitionen aus Data-Catalog-Indizes ab. Wird mit `push_down_predicate`, einer Option der the section called “create_dynamic_frame_from_catalog”-Methode (und anderen ähnlichen Methoden) verwendet. Weitere Informationen finden Sie unter the section called “Katalogpartitionsprädikate”.
- `"partitionKeys"`: (Optional) Wird zum Schreiben verwendet. Ein Array von Zeichenketten mit Spaltenbeschriftungen. AWS Glue partitioniert Ihre Daten wie in dieser Konfiguration angegeben. Weitere Informationen finden Sie unter the section called “Schreiben von Partitionen”.
- `"excludeStorageClasses"`: (Optional) Wird zum Lesen verwendet. Ein Array von Zeichenketten, die Amazon S3 S3-Speicherklassen angeben. AWS Glue schließt Amazon S3 S3-Objekte basierend auf dieser Konfiguration aus. Weitere Informationen finden Sie unter the section called “Ausschließen von Amazon-S3-Speicherklassen”.

Veraltete Verbindungssyntaxen für Datenformate

Auf bestimmte Datenformate kann mithilfe einer bestimmten Verbindungstyp-Syntax zugegriffen werden. Diese Syntax ist veraltet. Wir empfehlen Ihnen, Ihre Formate stattdessen mithilfe des `s3`-Verbindungstyps und der in the section called “Pfad-Formatoptionen” bereitgestellten Formatoptionen anzugeben.

"connectionType": "orc"

Bezeichnet eine Verbindung zu Dateien, die in Amazon S3 im Dateiformat Apache Hive Optimized Row Columnar (ORC) gespeichert sind.

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "orc"`:

- `paths`: (Erforderlich) Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- (weitere Name-Wert-Paare für Optionen): Alle zusätzlichen Optionen einschließlich Formatierungsoptionen werden direkt an die SparkSQL-`DataSource` übergeben.

"connectionType": "parquet"

Bezeichnet eine Verbindung mit Dateien, die in Amazon S3 im Dateiformat Apache Parquet gespeichert sind.

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "parquet"`:

- `paths`: (Erforderlich) Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- (weitere Name-Wert-Paare für Optionen): Alle zusätzlichen Optionen einschließlich Formatierungsoptionen werden direkt an die SparkSQL-`DataSource` übergeben.

Ausschließen von Amazon-S3-Speicherklassen

Wenn du rennst AWS Glue Bei ETL-Jobs, die Dateien oder Partitionen aus Amazon Simple Storage Service (Amazon S3) lesen, können Sie einige Amazon S3-Speicherklassentypen ausschließen.

Die folgenden Speicherklassen sind in Amazon S3 verfügbar:

- `STANDARD` – Für die universelle Speicherung häufig aufgerufener Daten.
- `INTELLIGENT_TIERING` – Für Daten mit wechselnden oder unbekannten Zugriffsmustern.
- `STANDARD_IA` und `ONEZONE_IA` – Für langlebige Daten, auf die weniger häufig zugegriffen wird.
- `GLACIER`, `DEEP_ARCHIVE` und `REDUCED_REDUNDANCY` – Für die Langzeitarchivierung und digitale Archivierung.

Weitere Informationen finden Sie unter Amazon S3 Storage Classes (Amazon-S3-Speicherklassen) im Amazon-S3-Entwicklerhandbuch.

Die Beispiele in diesem Abschnitt zeigen, wie Sie die `GLACIER`- und `DEEP_ARCHIVE`-Speicherklassen ausschließen. Mit diesen Klassen können Sie Dateien auflisten, aber Sie können die Dateien nur lesen, wenn sie wiederhergestellt werden. (Weitere Informationen finden Sie unter Restoring Archived Objects (Wiederherstellen archivierter Objekte) im Amazon-S3-Entwicklerhandbuch.)

Durch die Verwendung von Speicherklassenausschlüssen können Sie sicherstellen, dass AWS Glue Jobs funktionieren auf Tabellen, die über Partitionen in diesen Speicherklassenstufen verfügen. Ohne Ausnahmen schlagen Aufträge, die Daten aus diesen Stufen lesen, mit dem folgenden Fehler fehl: AmazonS3Exception: The operation is not valid for the object storage class.

Es gibt verschiedene Möglichkeiten, Amazon S3 S3-Speicherklassen zu filtern AWS Glue.

Themen

- Ausschließen von Amazon-S3-Speicherklassen beim Erstellen eines Dynamic Frames
- Ausschließen von Amazon-S3-Speicherklassen in einer Data-Catalog-Tabelle

Ausschließen von Amazon-S3-Speicherklassen beim Erstellen eines Dynamic Frames

Um Amazon S3 S3-Speicherklassen bei der Erstellung eines dynamischen Frames auszuschließen, verwenden Sie `excludeStorageClasses` in`additionalOptions`. AWS Glue verwendet automatisch seine eigene Amazon S3 `Lister` S3-Implementierung, um Dateien aufzulisten und auszuschließen, die den angegebenen Speicherklassen entsprechen.

Die folgenden Python- und Scala-Beispiele zeigen, wie Sie die Speicherklassen `GLACIER` und `DEEP_ARCHIVE` beim Erstellen eines Dynamic Frames ausschließen.

Python-Beispiel:

```
glueContext.create_dynamic_frame.from_catalog(
    database = "my_database",
    tableName = "my_table_name",
    redshift_tmp_dir = "",
    transformation_ctx = "my_transformation_context",
    additional_options = {
        "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)
```

Scala-Beispiel:

```
val* *df = glueContext.getCatalogSource(
    nameSpace, tableName, "", "my_transformation_context",
    additionalOptions = JsonOptions(
        Map("excludeStorageClasses" -> List("GLACIER", "DEEP_ARCHIVE"))
    )
).getDynamicFrame()
```

Ausschließen von Amazon-S3-Speicherklassen in einer Data-Catalog-Tabelle

Sie können Ausnahmen für Speicherklassen angeben, die von einem verwendet werden sollen AWS Glue ETL-Job als Tabellenparameter im AWS Glue-Datenkatalog. Sie können diesen Parameter mithilfe von AWS Command Line Interface (AWS CLI) oder programmgesteuert mithilfe der API in den `CreateTable` Vorgang einbeziehen. Weitere Informationen finden Sie unter Tabellenstruktur und. CreateTable

Sie können auch ausgeschlossene Speicherklassen auf der AWS Glue console.

Ausschließen von Amazon-S3-Speicherklassen (Konsole)

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole bei https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich auf der linken Seite Tabellen aus.
3. Wählen Sie den Tabellennamen in der Liste aus und klicken Sie dann auf Edit table (Tabelle bearbeiten).
4. Fügen Sie unter Table properties (Tabelleneigenschaften) **`excludeStorageClasses`** als Schlüssel und **`[\"GLACIER\",\"DEEP_ARCHIVE\"]`** als Wert hinzu.
5. Wählen Sie Anwenden aus.

Partitionen für die ETL-Ausgabe verwalten in AWS Glue

Die Partitionierung ist eine wichtige Methode zum Organisieren von Datasets, sodass diese effizient abgefragt werden können. Sie ordnet Daten basierend auf den individuellen Werten einer oder mehrerer Spalten in eine hierarchische Verzeichnisstruktur.

Beispiel: Sie können Ihre Anwendungsprotokolle in Amazon Simple Storage Service (Amazon S3) nach Datum, aufgeschlüsselt nach Jahr, Monat und Tag, partitionieren. Dateien, die der Datenmenge eines einzelnen Tages entsprechen, werden dann unter einem Präfix wie `s3://my_bucket/logs/year=2018/month=01/day=23/` geordnet. Systeme wie Amazon Athena, Amazon Redshift

Spectrum und jetzt AWS Glue kann diese Partitionen verwenden, um Daten nach Partitionswert zu filtern, ohne alle zugrunde liegenden Daten aus Amazon S3 lesen zu müssen.

Crawler leiten nicht nur Dateitypen und Schemas ab, sie identifizieren auch automatisch die Partitionsstruktur Ihres Datensatzes, wenn sie den AWS Glue-Datenkatalog füllen. Die resultierenden Partitionsspalten stehen für Abfragen zur Verfügung AWS Glue ETL-Jobs oder Abfrage-Engines wie Amazon Athena.

Nachdem Sie eine Tabelle durchsucht haben, können Sie die Partitionen anzeigen, die vom Crawler erstellt wurden. Im AWS Glue Konsole, wählen Sie im linken Navigationsbereich Tabellen aus. Wählen Sie die vom Crawler erstellte Tabelle und dann View Partitions (Partionen anzeigen) aus.

Für Apache Hive-ähnliche partitionierte Pfade im `key=val`-Stil füllen Crawler den Spaltennamen automatisch mit dem Schlüsselnamen. Andernfalls werden Standardnamen wie z. B. `partition_0`, `partition_1` usw. verwendet. Sie können die Standardnamen in der Konsole ändern. Navigieren Sie dazu zur Tabelle. Überprüfen Sie, ob Indizes auf der Registerkarte Indizes vorhanden sind. Wenn das der Fall ist, müssen Sie sie löschen, um fortzufahren (Sie können sie anschließend mit den neuen Spaltennamen neu erstellen). Wählen Sie dann Schema bearbeiten und ändern Sie dort die Namen der Partitionsspalten.

In Ihren ETL-Skripts können Sie dann die Partitionsspalten filtern. Da die Partitionsinformationen im Data Catalog gespeichert sind, verwenden Sie die `from_catalog`-API-Aufrufe, um die Partitionsspalten in `DynamicFrame` zu erfassen. Verwenden Sie z. B. `create_dynamic_frame.from_catalog` statt `create_dynamic_frame.from_options`.

Partitionierung ist eine Optimierungsmethode, die zu einer geringeren Menge an gescannten Daten führt. Weitere Informationen darüber, wie Sie feststellen können, wann diese Technik geeignet ist, finden Sie unter Reduce the amount of data scan im Leitfaden Best Practices for Performance Tuning AWS Glue for Apache Spark Jobs auf AWS Prescriptive Guidance.

Vorabfilterung mit Pushdown-Prädikaten

In vielen Fällen können Sie ein Pushdown-Prädikat zum Filtern von Partitionen verwenden, ohne alle Dateien Ihres Datasets auflisten und lesen zu müssen. Anstatt den gesamten Datensatz zu lesen und dann in einem zu filtern DynamicFrame, können Sie den Filter direkt auf die Metadaten der Partition im Datenkatalog anwenden. Dann listen Sie nur auf und lesen in eine ein, was Sie tatsächlich benötigen DynamicFrame.

In Python können Sie beispielsweise Folgendes schreiben:

```
glue_context.create_dynamic_frame.from_catalog(
    database = "my_S3_data_set",
    table_name = "catalog_data_table",
    push_down_predicate = my_partition_predicate)
```

Dadurch wird eine erstellt DynamicFrame , die nur die Partitionen im Datenkatalog lädt, die dem Prädikatausdruck entsprechen. Je nachdem, wie klein eine Teilmenge der Daten ist, die Sie laden, kann dies sehr viel Verarbeitungszeit sparen.

Der Prädikatausdruck kann ein beliebiger boolescher Ausdruck sein, der von Spark SQL unterstützt wird. Alles, was Sie in eine `WHERE`-Klausel einer Spark SQL-Abfrage aufnehmen können, wird unterstützt. Beispiel: Mit dem Prädikatausdruck `pushDownPredicate = "(year=='2017' and month=='04')"` werden nur die Partitionen in den Data Catalog geladen, die sowohl `year` gleich 2017 und `month` gleich 04 ist. Weitere Informationen finden Sie in der Apache Spark SQL-Dokumentation und insbesondere in der Scala SQL-Funktionsreferenz.

Serverseitige Filterung mit Katalogpartitionsprädikaten

Die Option `push_down_predicate` wird angewendet, nachdem alle Partitionen aus dem Katalog aufgelistet wurden und bevor Dateien von Amazon S3 für diese Partitionen angeboten werden. Wenn Sie viele Partitionen für eine Tabelle haben, kann die Liste der Katalogpartitionen immer noch zusätzlichen Zeitaufwand verursachen. Um diesen Mehraufwand zu beheben, können Sie serverseitiges Partitionsbereinigen mit der `catalogPartitionPredicate` Option verwenden, die Partitionsindizes im AWS Glue-Datenkatalog verwendet. Dies macht die Partitionsfilterung viel schneller, wenn Millionen von Partitionen in einer Tabelle vorhanden sind. Sie können sowohl `push_down_predicate` und `catalogPartitionPredicate` in `additional_options` zusammen verwenden, wenn Ihr `catalogPartitionPredicate` eine Prädikatsyntax erfordert, die noch nicht von den Katalogpartition-Indizes unterstützt wird.

Python:

```
dynamic_frame = glueContext.create_dynamic_frame.from_catalog(
    database=dbname,
    table_name=tablename,
    transformation_ctx="datasource0",
    push_down_predicate="day>=10 and customer_id like '10%'",
    additional_options={"catalogPartitionPredicate":"year='2021' and month='06'"}
)
```

Scala:

```
val dynamicFrame = glueContext.getCatalogSource(
    database = dbname,
    tableName = tablename,
    transformationContext = "datasource0",
    pushDownPredicate="day>=10 and customer_id like '10%'",
    additionalOptions = JsonOptions("""{
        "catalogPartitionPredicate": "year='2021' and month='06'"}""")
    ).getDynamicFrame()
```

> **Note**
>
> `push_down_predicate` und `catalogPartitionPredicate` verwenden verschiedene Syntaxen. Erstere verwendet die Spark-SQL-Standardsyntax und letztere verwendet den JSQL-Parser.

Schreiben von Partitionen

Standardmäßig DynamicFrame ist a beim Schreiben nicht partitioniert. Alle Ausgabedateien werden auf der obersten Ebene des angegebenen Ausgabepfads geschrieben. Bis vor Kurzem bestand die einzige Möglichkeit, eine DynamicFrame in Partitionen zu schreiben, darin, sie DataFrame vor dem Schreiben in eine Spark-SQL zu konvertieren.

Unterstützt DynamicFrames jetzt jedoch die native Partitionierung mit einer Tastenfolge, indem Sie die `partitionKeys` Option verwenden, wenn Sie eine Senke erstellen. Der folgende Python-Code schreibt beispielsweise in Amazon S3 einen Datensatz im Parquet-Format in Verzeichnisse, die nach dem Type-Feld partitioniert sind. Von dort aus können Sie diese Partitionen mit anderen Systemen wie Amazon Athena verarbeiten.

```
glue_context.write_dynamic_frame.from_options(
    frame = projectedEvents,
    connection_type = "s3",
    connection_options = {"path": "$outpath", "partitionKeys": ["type"]},
    format = "parquet")
```

Zusammenfassen von Eingabedateien in größeren Gruppen beim Lesen

Sie können Eigenschaften Ihrer Tabellen festlegen, um eine zu aktivieren AWS Glue ETL-Job zum Gruppieren von Dateien, wenn sie aus einem Amazon S3 S3-Datenspeicher gelesen werden. Diese Eigenschaften ermöglichen den einzelnen ETL-Aufträgen eine Gruppe von Eingabedateien

in eine einzelne In-Memory-Partition zu lesen. Dies ist besonders nützlich, wenn es eine große Anzahl kleiner Dateien in Ihrem Amazon-S3-Datenspeicher gibt. Wenn Sie bestimmte Eigenschaften festlegen, weisen Sie AWS Glue um Dateien innerhalb einer Amazon S3 S3-Datenpartition zu gruppieren und die Größe der zu lesenden Gruppen festzulegen. Sie können diese Optionen auch festlegen, wenn Sie mit der `create_dynamic_frame.from_options`-Methode Daten aus dem Amazon-S3-Datenspeicher lesen.

Zum Aktivieren von Gruppierungsdateien für eine Tabelle legen Sie Schlüssel-Wert-Paare im Parameterfeld Ihrer Tabellenstruktur fest. Verwenden Sie eine JSON-Notation, um einen Wert für das Parameterfeld Ihrer Tabelle festzulegen. Weitere Informationen zum Bearbeiten der Eigenschaften einer Tabelle finden Sie unter Tabellendetails anzeigen und verwalten.

Sie können mit dieser Methode das Gruppieren von Tabellen im Data Catalog bei Amazon-S3-Datenspeichern aktivieren.

groupFiles

Setzen Sie groupFiles auf, `inPartition` um die Gruppierung von Dateien innerhalb einer Amazon S3 S3-Datenpartition zu aktivieren. AWS Glue aktiviert automatisch die Gruppierung, wenn mehr als 50.000 Eingabedateien vorhanden sind, wie im folgenden Beispiel.

```
'groupFiles': 'inPartition'
```

groupSize

Legen Sie groupSize auf die Zielgröße von Gruppen in Bytes fest. Die groupSize-Eigenschaft ist optional, wenn sie nicht angegeben wird. AWS Glue berechnet eine Größe, um alle CPU-Kerne im Cluster zu verwenden und gleichzeitig die Gesamtzahl der ETL-Aufgaben und speicherinternen Partitionen zu reduzieren.

Im Folgenden wird beispielsweise die Gruppengröße auf 1 MB festgelegt.

```
'groupSize': '1048576'
```

Beachten Sie, dass `groupsize` mit dem Ergebnis einer Berechnung eingestellt werden sollte. Beispiel: 1024 * 1024 = 1048576.

recurse

Legen Sie recurse (rekursiv) auf `True` fest, um Dateien in allen Unterverzeichnissen zu lesen, wenn `paths` als ein Array von Pfaden angegeben wird. Sie müssen rekursiv nicht festlegen, wenn es sich bei `paths` um ein Array von Objektschlüsseln in Amazon S3 handelt, oder das Eingabeformat parquet/orc ist, wie im folgenden Beispiel.

```
'recurse':True
```

Wenn Sie direkt aus Amazon S3 mit der `create_dynamic_frame.from_options`-Methode lesen, fügen Sie diese Verbindungsoptionen hinzu. Im Folgenden wird beispielsweise versucht, Dateien in Gruppen mit einer Größe von 1 MB zusammenzufassen.

```
df = glueContext.create_dynamic_frame.from_options("s3", {'paths': ["s3://s3path/"],
 'recurse':True, 'groupFiles': 'inPartition', 'groupSize': '1048576'}, format="json")
```

> **Note**
>
> `groupFiles`wird für die DynamicFrames Erstellung aus den folgenden Datenformaten unterstützt: csv, ion, GrokLog, json und xml. Diese Option wird für Avro, Parquet und Orc nicht unterstützt.

Amazon-VPC-Endpunkte für Amazon S3

Aus Sicherheitsgründen führen viele AWS Kunden ihre Anwendungen in einer Amazon Virtual Private Cloud Cloud-Umgebung (Amazon VPC) aus. Mit Amazon VPC können Sie EC2 Amazon-Instances in einer virtuellen privaten Cloud starten, die logisch von anderen Netzwerken — einschließlich dem öffentlichen Internet — isoliert ist. Mit einer Amazon VPC können Sie den zugehörigen IP-Adressbereich, die Subnetze, Routing-Tabellen, Netzwerk-Gateways und Sicherheitseinstellungen steuern.

> **Note**
>
> Wenn Sie Ihr AWS Konto nach dem 04.12.2013 erstellt haben, haben Sie in jeder Region bereits eine Standard-VPC. AWS Sie können sofort beginnen, Ihre Standard-VPC zu verwenden, eine zusätzliche Konfiguration ist nicht erforderlich.
> Weitere Informationen finden Sie unter Ihre Standard-VPC und -Subnetze im Amazon-VPC-Benutzerhandbuch.

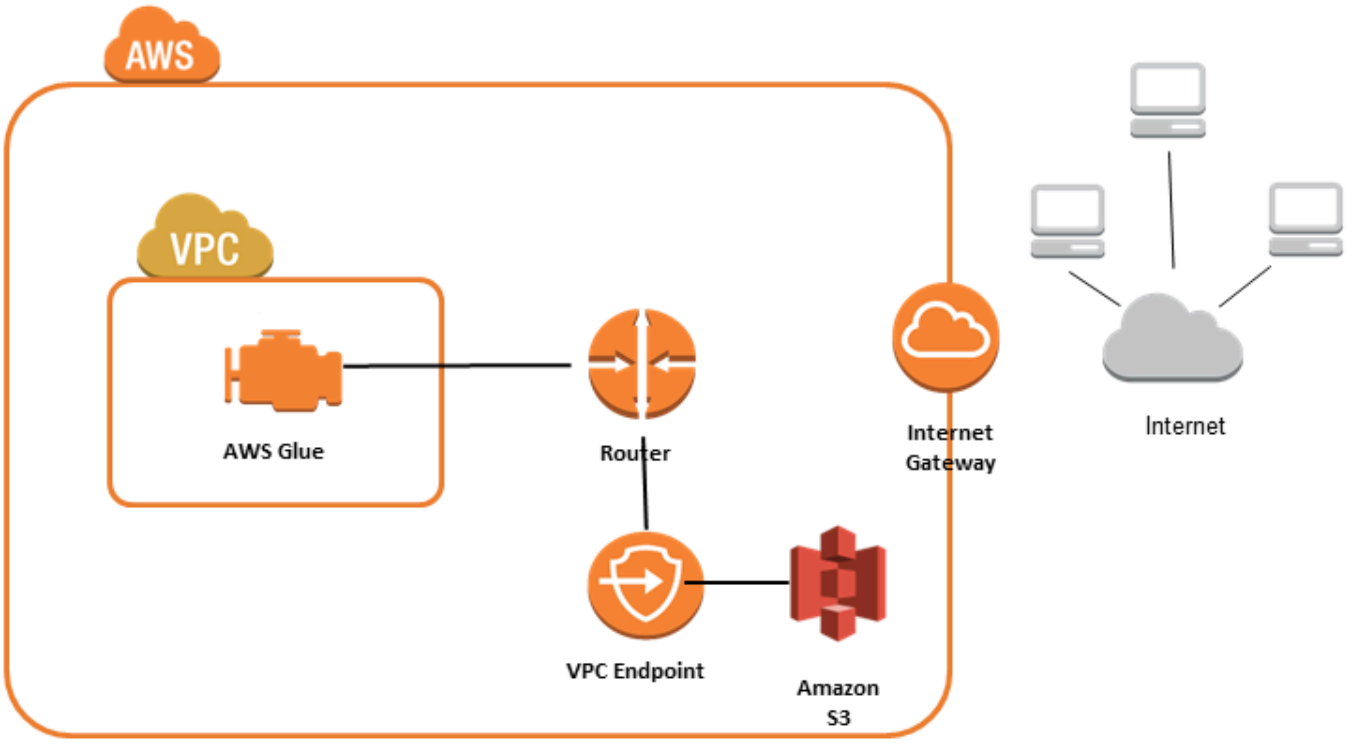
Viele Kunden haben legitime Bedenken hinsichtlich Datenschutz und Sicherheit, was das Senden und Empfangen von Daten über das öffentliche Internet angeht. Kunden können dieses Problem durch die Verwendung eines Virtual Private Network (VPN) lösen, um sämtlichen Amazon-S3-Netzwerkdatenverkehr über die eigene Unternehmensnetzwerkinfrastruktur zu leiten. Dieser Ansatz kann jedoch zu Herausforderungen hinsichtlich Bandbreite und Verfügbarkeit führen.

VPC-Endpunkte für Amazon S3 können diese Herausforderungen bewältigen helfen. Ein VPC-Endpunkt für Amazon S3 ermöglicht AWS Glue private IP-Adressen für den Zugriff auf Amazon S3 ohne Zugang zum öffentlichen Internet zu verwenden. AWS Glue benötigt keine öffentlichen IP-Adressen, und Sie benötigen kein Internet-Gateway, kein NAT-Gerät oder ein virtuelles privates Gateway in Ihrer VPC. Sie steuern den Zugriff auf Amazon S3 mittels Endpunktrichtlinien. Der Verkehr zwischen Ihrer VPC und dem AWS Service verlässt das Amazon-Netzwerk nicht.

Wenn Sie einen VPC-Endpunkt für Amazon S3 erstellen, werden alle Anfragen für einen Amazon-S3-Endpunkt in der Region (z. B. s3.us-west-2.amazonaws.com) an einen privaten Amazon-S3-Endpunkt innerhalb des Amazon-Netzwerks geleitet. Sie müssen Ihre Anwendungen, die auf EC2 Amazon-Instances in Ihrer VPC ausgeführt werden, nicht ändern — der Endpunktname bleibt derselbe, aber die Route zu Amazon S3 bleibt vollständig innerhalb des Amazon-Netzwerks und greift nicht auf das öffentliche Internet zu.

Weitere Informationen zu VPC-Endpunkten finden Sie unter VPC-Endpunkte im Amazon-VPC-Benutzerhandbuch.

Das folgende Diagramm zeigt, wie AWS Glue kann einen VPC-Endpunkt für den Zugriff auf Amazon S3 verwenden.



So richten Sie den Zugriff für Amazon S3 ein

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.
2. Wählen Sie im linken Navigationsbereich die Option Endpoints (Endpunkte) aus.
3. Wählen Sie Create Endpoint (Endpunkt erstellen) aus und befolgen Sie die Schritte zum Erstellen eines Amazon-S3-VPC-Endpunkts vom Typ Gateway.

Amazon-DocumentDB-Verbindungen

Sie können AWS Glue for Spark verwenden, um aus Tabellen in Amazon DocumentDB zu lesen und in Tabellen zu schreiben. Sie können eine Verbindung zu Amazon DocumentDB herstellen, indem

Sie Anmeldeinformationen verwenden, AWS Secrets Manager die über eine AWS Glue-Verbindung gespeichert sind.

Weitere Informationen zu Amazon DocumentDB finden Sie in der Dokumentation zu Amazon DocumentDB.

> **Note**
>
> Elastische Amazon DocumentDB-Cluster werden derzeit nicht unterstützt, wenn der AWS Glue-Connector verwendet wird. Weitere Informationen zu elastischen Clustern finden Sie unter Verwendung von elastischen Amazon-DocumentDB-Clustern.

Lesen und Schreiben in Amazon DocumentDB-Sammlungen

> **Note**
>
> Wenn Sie einen ETL-Auftrag erstellen, der eine Verbindung mit Amazon DocumentDB herstellt, müssen Sie für die `Connections`-Auftragseigenschaft ein Verbindungsobjekt festlegen, das die Virtual Private Cloud (VPC) angibt, in der Amazon DocumentDB ausgeführt wird. Beim Verbindungsobjekt muss der Verbindungstyp „`JDBC`“ und `JDBC URL` muss „`mongo://<DocumentDB_host>:27017`“ sein.

> **Note**
>
> Diese Codebeispiele wurden für AWS Glue 3.0 entwickelt. Informationen zur Migration auf AWS Glue 4.0 finden Sie unterthe section called “MongoDB”. Der `uri`-Parameter hat sich geändert.

> **Note**
>
> Wenn Sie Amazon DocumentDB verwenden, muss `retryWrites` in bestimmten Situationen auf „false“ gesetzt werden, z. B. wenn das geschriebene Dokument `_id` angibt. Weitere Informationen finden Sie unter Funktionsunterschiede zu MongoDB in der Dokumentation zu Amazon DocumentDB.

Das folgende Python-Skript veranschaulicht die Verwendung von Verbindungstypen und Verbindungsoptionen zum Lesen und Schreiben in Amazon DocumentDB.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)

output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
documentdb_uri = "mongodb://<mongo-instanced-ip-address>:27017"
documentdb_write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_docdb_options = {
    "uri": documentdb_uri,
    "database": "test",
    "collection": "coll",
    "username": "username",
    "password": "1234567890",
    "ssl": "true",
    "ssl.domain_match": "false",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"
}

write_documentdb_options = {
    "retryWrites": "false",
    "uri": documentdb_write_uri,
    "database": "test",
    "collection": "coll",
```

```
    "username": "username",
    "password": "pwd"
}

# Get DynamicFrame from  DocumentDB
dynamic_frame2 =
 glueContext.create_dynamic_frame.from_options(connection_type="documentdb",

 connection_options=read_docdb_options)

# Write DynamicFrame to MongoDB and DocumentDB
glueContext.write_dynamic_frame.from_options(dynamic_frame2,
 connection_type="documentdb",

 connection_options=write_documentdb_options)

job.commit()
```

Das folgende Scala-Skript veranschaulicht die Verwendung von Verbindungstypen und Verbindungsoptionen zum Lesen und Schreiben in Amazon DocumentDB.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DOC_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val DOC_WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val documentDBJsonOption = jsonOptions(DOC_URI)
  lazy val writeDocumentDBJsonOption = jsonOptions(DOC_WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from DocumentDB
```

```
    val resultFrame2: DynamicFrame = glueContext.getSource("documentdb",
 documentDBJsonOption).getDynamicFrame()

    // Write DynamicFrame to DocumentDB
    glueContext.getSink("documentdb", writeJsonOption).writeDynamicFrame(resultFrame2)

    Job.commit()
  }

  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"test",
         |"collection":"coll",
         |"username": "username",
         |"password": "pwd",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
         |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Referenz zur Amazon-DocumentDB-Verbindungsoption

Bezeichnet eine Verbindung zu Amazon DocumentDB (mit MongoDB-Kompatibilität).

Die Anschlussmöglichkeiten bei einer Quellverbindung und eine Senkenverbindung unterscheiden sich.

„connectionType“: „documentdb“ als Quelle

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "documentdb"` als Quelle:

- `"uri"`: (Erforderlich) Der als `mongodb://<host>:<port>` zu formatierende Amazon-DocumentDB-Host, aus dem gelesen werden soll.
- `"database"`: (Erforderlich) Die Amazon-DocumentDB-Datenbank, aus der gelesen werden soll.
- `"collection"`: (Erforderlich) Die Amazon-DocumentDB-Sammlung, aus der gelesen werden soll.
- `"username"`: (Erforderlich) Der Amazon-DocumentDB-Benutzername.

- `"password"`: (Erforderlich) Das Amazon-DocumentDB-Passwort.
- `"ssl"`: (Erforderlich bei Verwendung von SSL) Wenn Ihre Verbindung SSL verwendet, müssen Sie diese Option mit dem Wert `"true"` einfügen.
- `"ssl.domain_match"`: (Erforderlich bei Verwendung von SSL) Wenn Ihre Verbindung SSL verwendet, müssen Sie diese Option mit dem Wert `"false"` einfügen.
- `"batchSize"`: (Optional): Die Anzahl der pro Batch zurückzugebenden Dokumente, die innerhalb des Cursors interner Batches verwendet werden.
- `"partitioner"`: (Optional): Der Klassenname des Partitionierers zum Lesen von Eingabedaten aus Amazon DocumentDB. Der Konnektor stellt die folgenden Partitionierer bereit:
  - `MongoDefaultPartitioner`(Standard) (In AWS Glue 4.0 nicht unterstützt)
  - `MongoSamplePartitioner`(In AWS Glue 4.0 nicht unterstützt)
  - `MongoShardedPartitioner`
  - `MongoSplitVectorPartitioner`
  - `MongoPaginateByCountPartitioner`
  - `MongoPaginateBySizePartitioner`(In AWS Glue 4.0 nicht unterstützt)
- `"partitionerOptions"` (Optional): Optionen für den angegebenen Partitionierer. Die folgenden Optionen werden für jeden Partitionierer unterstützt:
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner`: `partitionKey`, partitionSizeMB
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner`: `partitionKey`, partitionSizeMB

  Weitere Informationen zu diesen Optionen finden Sie unter Partitioner-Konfiguration in der MongoDB-Dokumentation.

„connectionType“: „documentdb“ als Senke

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "documentdb"` als Senke:

- `"uri"`: (Erforderlich) Der als `mongodb://<host>:<port>` zu formatierende Amazon-DocumentDB-Host, in den geschrieben werden soll.
- `"database"`: (Erforderlich) Die Amazon-DocumentDB-Datenbank, in die geschrieben werden soll.

- `"collection"`: (Erforderlich) Die Amazon-DocumentDB-Sammlung, in die geschrieben werden soll.
- `"username"`: (Erforderlich) Der Amazon-DocumentDB-Benutzername.
- `"password"`: (Erforderlich) Das Amazon-DocumentDB-Passwort.
- `"extendedBsonTypes"`: (Optional) Wenn `true`, sind erweiterte BSON-Typen beim Schreiben von Daten in Amazon DocumentDB erlaubt. Der Standardwert ist `true`.
- `"replaceDocument"`: (Optional) Wenn `true`, wird das gesamte Dokument beim Speichern von Datasets, die ein `_id`-Feld enthalten, ersetzt. Wenn `false`, werden nur Felder im Dokument aktualisiert, die mit den Feldern im Dataset übereinstimmen. Der Standardwert ist `true`.
- `"maxBatchSize"`: (Optional): Die maximale Batchgröße für Massenvorgänge bei der Datenspeicherung. Der Standardwert ist 512.
- `"retryWrites"`: (Optional): Automatische Wiederholung bestimmter Schreibvorgänge ein einziges Mal, wenn AWS Glue auf einen Netzwerkfehler stößt.

OpenSearch Serviceanschlüsse

Sie können AWS Glue for Spark verwenden, um in OpenSearch Service in AWS Glue 4.0 und späteren Versionen aus Tabellen zu lesen und in Tabellen zu schreiben. Sie können mit einer OpenSearch Abfrage definieren, was aus OpenSearch Service gelesen werden soll. Sie stellen eine Verbindung mit dem OpenSearch Dienst her, indem Sie die Anmeldeinformationen für die HTTP-Standardauthentifizierung verwenden, AWS Secrets Manager die über eine AWS Glue-Verbindung gespeichert sind. Diese Funktion ist nicht mit OpenSearch Service Serverless kompatibel.

Weitere Informationen zu Amazon OpenSearch Service finden Sie in der Amazon OpenSearch Service-Dokumentation.

Konfiguration von OpenSearch Service-Verbindungen

Um von AWS Glue aus eine Verbindung zu OpenSearch Service herzustellen, müssen Sie Ihre OpenSearch Service-Anmeldeinformationen in einem AWS Secrets Manager Secret erstellen und speichern und dieses Secret dann mit einer OpenSearch Service AWS Glue-Verbindung verknüpfen.

Voraussetzungen:

- Identifizieren Sie den Domain-Endpunkt *`aosEndpoint`* und den Port, von dem *`aosPort`* Sie lesen möchten, oder erstellen Sie die Ressource, indem Sie den Anweisungen in der Amazon OpenSearch Service-Dokumentation folgen. Weitere Informationen zum Erstellen einer Domain

finden Sie unter Amazon OpenSearch Service-Domains erstellen und verwalten in der Amazon OpenSearch Service-Dokumentation.

Ein Amazon OpenSearch Service-Domain-Endpunkt hat das folgende Standardformat: https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com. Weitere Informationen zur Identifizierung Ihres Domain-Endpunkts finden Sie unter Amazon OpenSearch Service-Domains erstellen und verwalten in der Amazon OpenSearch Service-Dokumentation.

Identifizieren oder generieren Sie Anmeldeinformationen für die HTTP-Basisauthentifizierung *aosUser* und *aosPassword* für Ihre Domain.

Um eine Verbindung zum OpenSearch Service zu konfigurieren:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer OpenSearch Dienstanmeldedaten ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `USERNAME` mit dem Wert. *aosUser*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `PASSWORD` mit dem Wert. *aosPassword*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für die future Verwendung in AWS Glue bei. *connectionName*

   - Wählen Sie bei der Auswahl eines Verbindungstyps OpenSearch Service aus.
   - Geben Sie bei der Auswahl eines Domänenendpunkts Folgendes an*aosEndpoint*.
   - Geben Sie bei der Auswahl eines Ports an*aosPort*.
   - Geben Sie bei der Auswahl eines AWS Secrets Folgendes an*secretName*.

Nachdem Sie eine AWS OpenSearch Glue-Service-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue-Job ausführen:

- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *secretName* Leserechte.

- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus OpenSearch Dienstindizes lesen

Voraussetzungen:

- Ein OpenSearch Serviceindex, aus dem Sie lesen möchten,*aosIndex*.
- Eine AWS Glue OpenSearch Service-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen bereitstellt. Um dies zu erhalten, führen Sie die Schritte im vorherigen Verfahren, So konfigurieren Sie eine Verbindung zum OpenSearch Service, durch. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

In diesem Beispiel wird ein Index von Amazon OpenSearch Service gelesen. Sie müssen den Parameter `pushdown` angeben.

Zum Beispiel:

```
opensearch_read = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "pushdown": "true",
    }
)
```

Sie können auch eine Abfragezeichenfolge angeben, um die in Ihrem zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `opensearch.query` konfigurieren.

`opensearch.query`kann eine URL-Abfrageparameterzeichenfolge *queryString* oder ein DSL-JSON-Abfrageobjekt *queryObject* verwenden. Weitere Informationen zur DSL-Abfrage finden Sie in der Dokumentation unter Query-DSL. OpenSearch Um eine URL-Abfrageparameterzeichenfolge bereitzustellen, stellen Sie `?q=` Ihrer Abfrage wie bei einer vollqualifizierten URL voran. Um ein DSL-Abfrageobjekt bereitzustellen, versehen Sie das JSON-Objekt in Escape-Zeichen, bevor Sie es bereitstellen.

Zum Beispiel:

```
            queryObject = "{ "query": { "multi_match": { "query": "Sample", "fields":
 [ "sample" ] } } }"
            queryString = "?q=queryString"

            opensearch_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "opensearch.query": queryString,
        "pushdown": "true",
    }
)
```

Weitere Informationen zum Erstellen einer Abfrage außerhalb ihrer spezifischen Syntax finden Sie in der Dokumentation unter Syntax von Abfragezeichenfolgen. OpenSearch

Wenn Sie aus OpenSearch Sammlungen lesen, die Daten vom Typ Array enthalten, müssen Sie in Ihrem Methodenaufruf mithilfe des `opensearch.read.field.as.array.include` Parameters angeben, welche Felder vom Typ Array sind.

Wenn Sie beispielsweise das folgende Dokument lesen, stoßen Sie auf die Array-Felder `genre` und `actor`:

```
{
    "_index": "movies",
    "_id": "2",
    "_version": 1,
    "_seq_no": 0,
    "_primary_term": 1,
    "found": true,
    "_source": {
        "director": "Frankenheimer, John",
        "genre": [
            "Drama",
            "Mystery",
            "Thriller",
            "Crime"
        ],
        "year": 1962,
        "actor": [
```

```
            "Lansbury, Angela",
            "Sinatra, Frank",
            "Leigh, Janet",
            "Harvey, Laurence",
            "Silva, Henry",
            "Frees, Paul",
            "Gregory, James",
            "Bissell, Whit",
            "McGiver, John",
            "Parrish, Leslie",
            "Edwards, James",
            "Flowers, Bess",
            "Dhiegh, Khigh",
            "Payne, Julie",
            "Kleeb, Helen",
            "Gray, Joe",
            "Nalder, Reggie",
            "Stevens, Bert",
            "Masters, Michael",
            "Lowell, Tom"
        ],
        "title": "The Manchurian Candidate"
    }
}
```

In diesem Fall würden Sie diese Feldnamen in den Methodenaufruf aufnehmen. Zum Beispiel:

```
"opensearch.read.field.as.array.include": "genre,actor"
```

Wenn ein Array-Feld innerhalb der Dokumentstruktur verschachtelt ist, verweisen Sie darauf in Punktnotation: `"genre,actor,foo.bar.baz"`. Damit geben Sie das Array `baz` in Ihrem Quelldokument an, und zwar über das eingebettete Dokument `foo`, das das eingebettete Dokument `bar` enthält.

In OpenSearch Service-Tabellen schreiben

In diesem Beispiel werden Informationen aus einem vorhandenen DynamicFrame OpenSearch Dienst *`dynamicFrame`* in einen Dienst geschrieben. Wenn der Index bereits Informationen enthält, hängt AWS Glue Daten von Ihrem DynamicFrame an. Sie müssen den Parameter `pushdown` angeben.

Voraussetzungen:

- Eine OpenSearch Service-Tabelle, in die Sie schreiben möchten. Sie benötigen Identifikationsinformationen für die Tabelle. Nennen wir das*tableName*.
- Eine AWS Glue OpenSearch Service-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen bereitstellt. Um dies zu erhalten, führen Sie die Schritte im vorherigen Verfahren, So konfigurieren Sie eine Verbindung zum OpenSearch Service, durch. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="opensearch",
    connection_options={
      "connectionName": "connectionName",
      "opensearch.resource": "aosIndex",
    },
)
```

OpenSearch Referenz zur Dienstverbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue OpenSearch Service-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen für Ihre Verbindungsmethode bereitstellt.
- `opensearch.resource` – Erforderlich. Wird für Lesen/Schreiben verwendet. Gültige Werte: OpenSearch Indexnamen. Der Indexname, mit dem Ihre Verbindungsmethode interagieren wird.
- `opensearch.query` – Wird zum Lesen verwendet. Gültige Werte: JSON, in der eine Zeichenfolge in Escape-Zeichen stehen, oder, wenn diese Zeichenfolge mit ? beginnt, der Suchteil einer URL. Eine OpenSearch Abfrage, die filtert, was beim Lesen abgerufen werden soll. Weitere Informationen zur Verwendung dieses Parameters finden Sie im vorherigen Abschnitt the section called “Aus dem OpenSearch Service lesen”.
- `pushdown` – erforderlich. Wird zum Lesen verwendet. Zulässige Werte: Boolesch. Weist Spark an, Leseabfragen OpenSearch so weiterzuleiten, dass die Datenbank nur relevante Dokumente zurückgibt.
- `opensearch.read.field.as.array.include` – Erforderlich, wenn Daten vom Typ Array gelesen werden. Wird zum Lesen verwendet. Gültige Werte: durch Kommas getrennte Listen von Feldnamen. Gibt Felder an, die als Arrays aus OpenSearch Dokumenten gelesen werden sollen.

Weitere Informationen zur Verwendung dieses Parameters finden Sie im vorherigen Abschnitt the section called “Aus dem OpenSearch Service lesen”.

Redshift-Verbindungen

Sie können AWS Glue for Spark verwenden, um aus Tabellen in Amazon Redshift Redshift-Datenbanken zu lesen und in Tabellen zu schreiben. Bei der Verbindung zu Amazon Redshift-Datenbanken verschiebt AWS Glue Daten mithilfe von Amazon Redshift SQL `COPY` und `UNLOAD` Befehlen über Amazon S3, um einen maximalen Durchsatz zu erzielen. In AWS Glue 4.0 und höher können Sie die Amazon Redshift Redshift-Integration für Apache Spark verwenden, um mit Amazon Redshift Redshift-spezifischen Optimierungen und Funktionen zu lesen und zu schreiben, die über diejenigen hinausgehen, die bei Verbindungen über frühere Versionen verfügbar sind.

Erfahren Sie, wie AWS Glue es für Amazon Redshift Redshift-Benutzer einfacher denn je macht, für serverlose Datenintegration und ETL zu AWS Glue zu migrieren.

Konfigurieren von Redshift-Verbindungen

Um Amazon Redshift Redshift-Cluster in AWS Glue verwenden zu können, benötigen Sie einige Voraussetzungen:

- Ein Amazon-S3-Verzeichnis zur temporären Speicherung beim Lesen und Schreiben in die Datenbank.
- Eine Amazon VPC, die die Kommunikation zwischen Ihrem Amazon Redshift Redshift-Cluster, Ihrem AWS Glue-Job und Ihrem Amazon S3 S3-Verzeichnis ermöglicht.
- Entsprechende IAM-Berechtigungen für den AWS Glue-Job und den Amazon Redshift Redshift-Cluster.

Konfiguration von IAM-Rollen

Einrichten der Rolle für den Amazon-Redshift-Cluster

Ihr Amazon Redshift Redshift-Cluster muss in der Lage sein, in Amazon S3 zu lesen und in Amazon S3 zu schreiben, um in AWS Glue-Jobs integriert zu werden. Um dies zu ermöglichen, können Sie IAM-Rollen dem Amazon-Redshift-Cluster zuordnen, mit dem Sie eine Verbindung herstellen möchten. Ihre Rolle sollte über eine Richtlinie verfügen, die das Lesen und Schreiben in Ihrem temporären Amazon-S3-Verzeichnis ermöglicht. Ihre Rolle sollte über eine Vertrauensbeziehung verfügen, die den `redshift.amazonaws.com`-Service zu `AssumeRole` ermöglicht.

So ordnen Sie Amazon Redshift eine IAM-Rolle zu

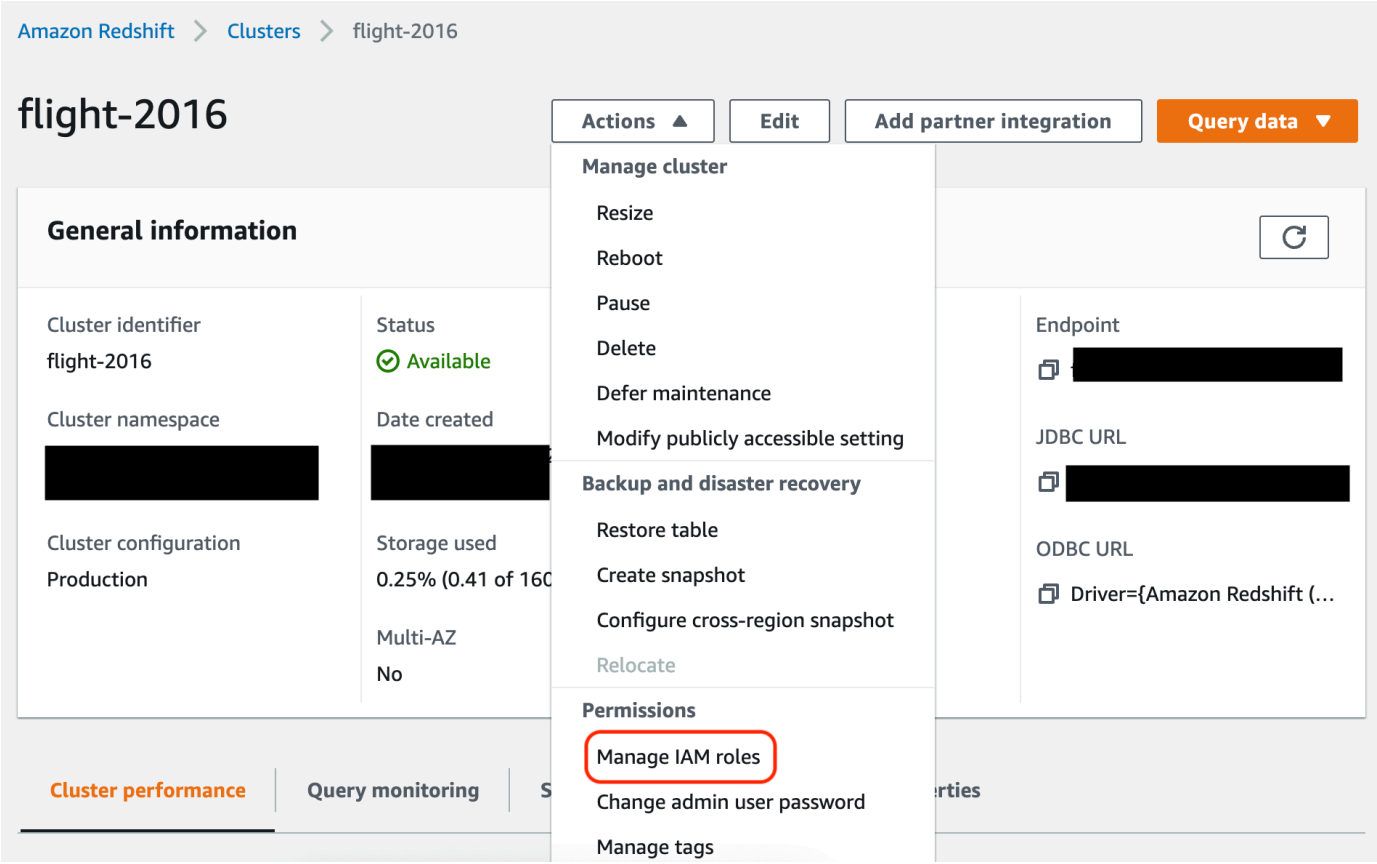
1. Voraussetzungen: Ein Amazon-S3-Bucket oder -Verzeichnis, der für die temporäre Speicherung von Dateien verwendet wird.
2. Identifizieren Sie, welche Amazon-S3-Berechtigungen Ihr Amazon-Redshift-Cluster benötigt. Beim Verschieben von Daten zu und von einem Amazon Redshift-Cluster geben AWS Glue-Jobs COPY- und UNLOAD-Anweisungen gegen Amazon Redshift aus. Wenn Ihr Job eine Tabelle in Amazon Redshift ändert, gibt AWS Glue auch CREATE LIBRARY-Anweisungen aus. Informationen zu bestimmten Amazon S3 S3-Berechtigungen, die Amazon Redshift zur Ausführung dieser Anweisungen benötigt, finden Sie in der Amazon Redshift-Dokumentation: Amazon Redshift: Zugriffsberechtigungen für andere Ressourcen. AWS
3. Erstellen Sie in der IAM-Konsole eine IAM-Richtlinie mit den erforderlichen Berechtigungen. Weitere Informationen zum Erstellen einer Richtlinie finden Sie unter Erstellen von IAM-Richtlinien.
4. Erstellen Sie in der IAM-Konsole eine Rolle und eine Vertrauensbeziehung, die es Amazon Redshift ermöglicht, die Rolle zu übernehmen. Folgen Sie den Anweisungen in der IAM-Dokumentation, um eine Rolle für einen AWS Service (Konsole) zu erstellen

   - Wenn Sie aufgefordert werden, einen AWS Service-Anwendungsfall auszuwählen, wählen Sie „Redshift — Customizable“.
   - Wenn Sie aufgefordert werden, eine Richtlinie anzufügen, wählen Sie die zuvor definierte Richtlinie aus.

   > **Note**
   >
   > Weitere Informationen zur Konfiguration von Rollen für Amazon Redshift finden Sie in der Amazon Redshift-Dokumentation unter Autorisieren von Amazon Redshift, in Ihrem Namen auf andere AWS Services zuzugreifen.

5. Ordnen Sie in der Amazon-Redshift-Konsole die Rolle Ihrem Amazon-Redshift-Cluster zu. Folgen Sie den Anweisungen in der Amazon-Redshift-Dokumentation.

   Wählen Sie die hervorgehobene Option in der Amazon-Redshift-Konsole aus, um diese Einstellung zu konfigurieren:



> **Note**
>
> Standardmäßig übergeben AWS Glue-Jobs temporäre Amazon Redshift Redshift-Anmeldeinformationen, die mit der Rolle erstellt wurden, die Sie für die Ausführung des Jobs angegeben haben. Wir raten von der Verwendung dieser Anmeldeinformationen ab. Aus Sicherheitsgründen verfallen diese Zugangsdaten nach einer Stunde.

Richten Sie die Rolle für den AWS Glue-Job ein

Der AWS Glue-Job benötigt eine Rolle für den Zugriff auf den Amazon S3 S3-Bucket. Sie benötigen keine IAM-Berechtigungen für den Amazon-Redshift-Cluster. Ihr Zugriff wird durch die Konnektivität in Amazon VPC und Ihre Datenbankanmeldeinformationen gesteuert.

Einrichten der Amazon VPC

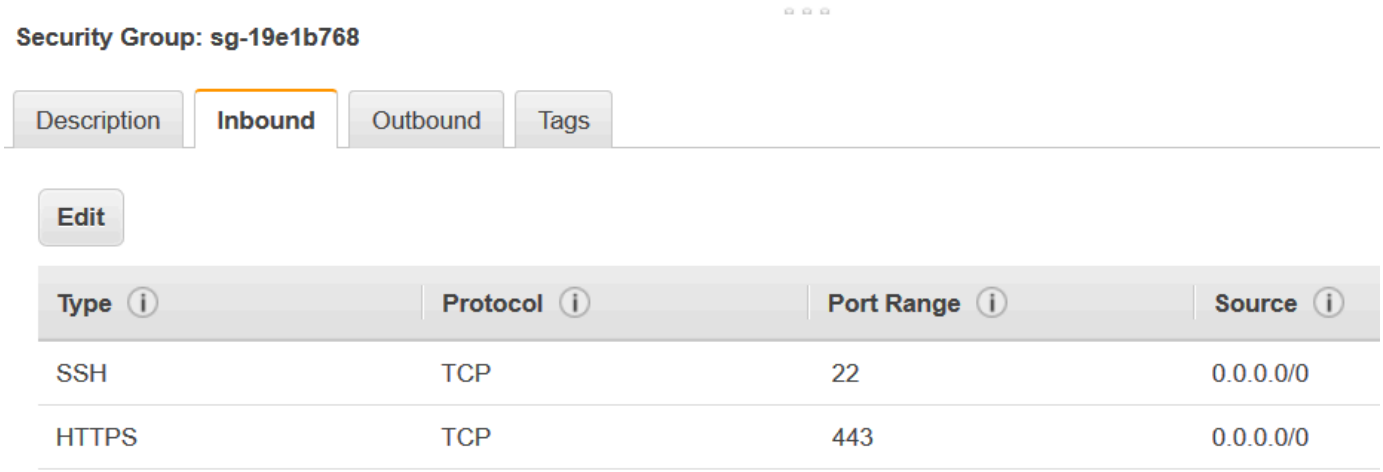
So richten Sie den Zugriff für Amazon-Redshift-Datenspeicher ein

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon Redshift Redshift-Konsole unter https://console.aws.amazon.com/redshiftv2/.
2. Wählen Sie im linken Navigationsbereich Cluster aus.
3. Wählen Sie den Cluster-Namen aus, auf den Sie von AWS Glue zugreifen möchten.
4. Wählen Sie im Abschnitt Cluster Properties (Cluster-Eigenschaften) eine Sicherheitsgruppe in VPC-Sicherheitsgruppen aus, die AWS Glue verwenden darf. Notieren Sie sich den Namen der Sicherheitsgruppe, die Sie gewählt haben, um später darauf zurückgreifen zu können. Wenn Sie die Sicherheitsgruppe auswählen, wird die Liste der Sicherheitsgruppen der EC2 Amazon-Konsole geöffnet.
5. Wählen Sie die Sicherheitsgruppe aus, die geändert werden soll, und navigieren Sie zur Registerkarte Eingehend.
6. Fügen Sie eine selbstreferenzierende Regel hinzu, um die Kommunikation von AWS Glue-Komponenten zuzulassen. Insbesondere fügen Sie hinzu oder bestätigen Sie, dass eine Regel des Typs `All TCP` vorhanden ist, das Protokoll `TCP` lautet, der Port-Bereich alle Ports umfasst und deren Quelle über denselben Sicherheitsgruppennamen verfügt wie die Gruppen-ID.

   Die eingehende Regel sollte wie folgt aussehen:

Typ	Protocol (Protokoll)	Port-Bereich	Quelle
Alle TCP	TCP	0–65535	database-security-group

   Zum Beispiel:



7. Fügen Sie ebenfalls eine Regel für ausgehenden Datenverkehr hinzu. Erlauben Sie entweder ausgehenden Datenverkehr für alle Ports, beispielsweise:

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Gesamter Datenverkehr	ALL	ALL	0.0.0.0/0

Oder erstellen Sie eine selbstreferenzierende Regel, wobei der Typ `All TCP` ist, das Protokoll `TCP` lautet, der Port-Bereich alle Ports umfasst und deren Ziel über denselben Sicherheitsgruppennamen wie die Gruppen-ID verfügt. Wenn Sie einen Amazon-S3-VPC-Endpunkt verwenden, fügen Sie auch eine HTTPS-Regel für den Amazon-S3-Zugriff hinzu. Das *`s3-prefix-list-id`* ist in der Sicherheitsgruppenregel erforderlich, um Datenverkehr von der VPC zum Amazon S3 S3-VPC-Endpunkt zuzulassen.

Zum Beispiel:

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Alle TCP	TCP	0–65535	*`security-group`*
HTTPS	TCP	443	*`s3-prefix-list-id`*

AWS Glue einrichten

Sie müssen eine AWS Glue Data Catalog-Verbindung erstellen, die Amazon VPC-Verbindungsinformationen bereitstellt.

So konfigurieren Sie die Amazon Redshift Amazon VPC-Konnektivität zu AWS Glue in der Konsole

1. Erstellen Sie eine Data-Catalog-Verbindung, indem Sie die Schritte in the section called “AWS Glue Verbindung hinzufügen” ausführen. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *connectionName*

   - Wählen Sie bei der Auswahl eines Verbindungstyps die Option Amazon Redshift aus.
   - Wählen Sie bei der Auswahl eines Redshift-Clusters Ihren Cluster nach Namen aus.
   - Stellen Sie Standardverbindungsinformationen für einen Amazon-Redshift-Benutzer in Ihrem Cluster bereit.
   - Ihre Amazon-VPC-Einstellungen werden automatisch konfiguriert.

   > **Note**
   >
   > Sie müssen `PhysicalConnectionRequirements` manuell für Ihre Amazon VPC bereitstellen, wenn Sie eine Amazon Redshift-Verbindung über das AWS SDK erstellen.

2. Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Beispiel: Lesen aus Amazon-Redshift-Tabellen

Sie können aus Amazon-Redshift-Clustern und Serverless-Amazon-Redshift-Umgebungen lesen.

Voraussetzungen: Eine Amazon-Redshift-Tabelle, aus der Sie gerne lesen möchten. Folgen Sie den Schritten im vorherigen Abschnitt. the section called “Redshift konfigurieren” Danach sollten Sie den Amazon S3 S3-URI für ein temporäres Verzeichnis *temp-s3-dir* und eine IAM-Rolle,*rs-role-name*, (im Konto*role-account-id*) haben.

Using the Data Catalog

Zusätzliche Voraussetzungen: Eine Data-Catalog-Datenbank und -Tabelle für die Amazon-Redshift-Tabelle, aus der Sie lesen möchten. Weitere Informationen zu Data Catalog finden Sie

unter Datenermittlung und Katalogisierung. Nachdem Sie einen Eintrag für Ihre Amazon Redshift Redshift-Tabelle erstellt haben, identifizieren Sie Ihre Verbindung mit einem *redshift-dc-database-name* und*redshift-table-name*.

Konfiguration: In Ihren Funktionsoptionen identifizieren Sie Ihre Datenkatalogtabelle mit den `database`- und `table_name`-Parametern. Sie identifizieren Ihr temporäres Amazon-S3-Verzeichnis mit `redshift_tmp_dir`. Sie geben dies auch *rs-role-name* mithilfe des `aws_iam_role` Schlüssels im `additional_options` Parameter an.

```
 glueContext.create_dynamic_frame.from_catalog(
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::role-account-id:role/rs-
role-name"})
```

Connecting directly

Zusätzliche Voraussetzungen: Sie benötigen den Namen Ihrer Amazon Redshift Redshift-Tabelle (*redshift-table-name*. Sie benötigen die JDBC-Verbindungsinformationen für den Amazon-Redshift-Cluster, der diese Tabelle speichert. Sie geben Ihre Verbindungsinformationen mit*host*, *portredshift-database-name*, *username* und *password* an.

Sie können Ihre Verbindungsinformationen von der Amazon-Redshift-Konsole abrufen, wenn Sie mit Amazon-Redshift-Clustern arbeiten. Wenn Sie Amazon Redshift Serverless verwenden, lesen Sie den Abschnitt Herstellen einer Verbindung zu Amazon Redshift Serverless in der Amazon Redshift-Dokumentation.

Konfiguration: In Ihren Funktionsoptionen kennzeichnen Sie Ihre Verbindungsparameter `url`, `dbtable`, `user` und `password`. Sie identifizieren Ihr temporäres Amazon-S3-Verzeichnis mit `redshift_tmp_dir`. Sie können Ihre IAM-Rolle mit `aws_iam_role` angeben, wenn Sie `from_options` verwenden. Die Syntax ähnelt der Verbindung über den Datenkatalog, Sie geben die Parameter jedoch in die `connection_options`-Zuordnung ein.

Es ist eine schlechte Praxis, Passwörter in AWS Glue-Skripten fest zu codieren. Erwägen Sie, Ihre Passwörter in Ihrem Skript mit SDK for Python (Boto3) zu speichern AWS Secrets Manager und von dort abzurufen.

```
my_conn_options = {
    "url": "jdbc:redshift://host:port/redshift-database-name",
    "dbtable": "redshift-table-name",
    "user": "username",
    "password": "password",
    "redshiftTmpDir": args["temp-s3-dir"],
    "aws_iam_role": "arn:aws:iam::account id:role/rs-role-name"
}

df = glueContext.create_dynamic_frame.from_options("redshift", my_conn_options)
```

Beispiel: Schreiben in Amazon-Redshift-Tabellen

Sie können in Amazon-Redshift-Cluster und Serverless-Amazon-Redshift-Umgebungen schreiben.

Voraussetzungen: Ein Amazon Redshift Redshift-Cluster und folgen Sie den Schritten im vorherigen Abschnitt. the section called “Redshift konfigurieren” Danach sollten Sie die Amazon S3 S3-URI für ein temporäres Verzeichnis *temp-s3-dir* und eine IAM-Rolle,*rs-role-name*, (im Konto*role-account-id*) haben. Sie benötigen außerdem einen `DynamicFrame`, dessen Inhalt Sie in die Datenbank schreiben möchten.

Using the Data Catalog

Zusätzliche Voraussetzungen Eine Datenkatalogdatenbank für den Amazon-Redshift-Cluster und die Tabelle, in die Sie schreiben möchten. Weitere Informationen zu Data Catalog finden Sie unter Datenermittlung und Katalogisierung. Sie identifizieren Ihre Verbindung mit *redshift-dc-database-name* und die Zieltabelle mit. *redshift-table-name*

Konfiguration: In Ihren Funktionsoptionen identifizieren Sie Ihre Data-Catalog-Datenbank mit dem `database`-Parameter und stellen dann die Tabelle mit `table_name` bereit. Sie identifizieren Ihr temporäres Amazon-S3-Verzeichnis mit `redshift_tmp_dir`. Sie werden dies auch *rs-role-name* mithilfe des `aws_iam_role` Schlüssels im `additional_options` Parameter angeben.

```
glueContext.write_dynamic_frame.from_catalog(
   frame = input dynamic frame,
   database = "redshift-dc-database-name",
   table_name = "redshift-table-name",
```

```
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::account-id:role/rs-role-
name"})
```

Connecting through a AWS Glue connection

Mit dieser `write_dynamic_frame.from_options`-Methode können Sie direkt eine Verbindung zu Amazon Redshift herstellen. Anstatt Ihre Verbindungsdetails jedoch direkt in Ihr Skript einzufügen, können Sie mit der `from_jdbc_conf`-Methode auf Verbindungsdetails verweisen, die in einer Data-Catalog-Verbindung gespeichert sind. Sie können dies tun, ohne Ihre Datenbank zu crawlen oder Data-Catalog-Tabellen zu erstellen. Weitere Informationen zu Data-Catalog-Verbindungen finden Sie unter Herstellen einer Verbindung zu Daten.

Zusätzliche Voraussetzungen: Eine Data-Catalog-Verbindung für Ihre Datenbank, eine Amazon-Redshift-Tabelle, aus der Sie lesen möchten

Konfiguration: Sie identifizieren Ihre Datenkatalogverbindung mit*dc-connection-name*. Sie identifizieren Ihre Amazon Redshift Redshift-Datenbank und -Tabelle mit *redshift-table-name* und*redshift-database-name*. Sie geben Ihre Data Catalog-Verbindungsinformationen mit `catalog_connection` und Ihre Amazon-Redshift-Informationen mit `dbtable` und `database` an. Die Syntax ähnelt der Verbindung über den Datenkatalog, Sie geben die Parameter jedoch in die `connection_options`-Zuordnung ein.

```
my_conn_options = {
    "dbtable": "redshift-table-name",
    "database": "redshift-database-name",
    "aws_iam_role": "arn:aws:iam::role-account-id:role/rs-role-name"
}

glueContext.write_dynamic_frame.from_jdbc_conf(
    frame = input dynamic frame,
    catalog_connection = "dc-connection-name",
    connection_options = my_conn_options,
    redshift_tmp_dir = args["temp-s3-dir"])
```

Referenz zur Amazon-Redshift-Verbindungsoption

Die grundlegenden Verbindungsoptionen, die für alle AWS Glue-JDBC-Verbindungen verwendet werden, um Informationen wie einzurichten`url`, `user` und `password` sind für alle JDBC-Typen konsistent. Weitere Informationen zu Standard-JDBC-Parametern finden Sie unter the section called “JDBC-Verbindungsparameter”.

Der Amazon-Redshift-Verbindungstyp erfordert einige zusätzliche Verbindungsoptionen:

- `"redshiftTmpDir"`: (Erforderlich) Der Amazon-S3-Pfad, in dem temporäre Daten beim Kopieren aus der Datenbank bereitgestellt werden können.
- `"aws_iam_role"`: (Optional) ARN für eine IAM-Rolle. Der AWS Glue-Job leitet diese Rolle an den Amazon Redshift Redshift-Cluster weiter, um den Cluster-Berechtigungen zu erteilen, die für die Ausführung der Anweisungen aus dem Job erforderlich sind.

Zusätzliche Verbindungsoptionen in AWS Glue 4.0+ verfügbar

Sie können Optionen für den neuen Amazon Redshift Redshift-Connector auch über die AWS Glue-Verbindungsoptionen übergeben. Eine vollständige Liste der unterstützten Konnektor-Optionen finden Sie im Abschnitt Spark-SQL-Parameter unter Amazon-Redshift-Integration für Apache Spark.

Zur Vereinfachung möchten wir Sie hier noch einmal auf bestimmte neue Optionen hinweisen:

Name	Erforderlich	Standard	Beschreibung
autopushdown	Nein	TRUE	Wendet Prädikat- und Abfrage-Pushdown durch Erfassen und Analysieren der logischen Spark-Plä ne für SQL-Vorgänge an. Die Operationen werden in eine SQL-Abfrage übersetzt und dann zur Verbesser ung der Leistung in Amazon Redshift ausgeführt.

Name	Erforderlich	Standard	Beschreibung
autopushdown.s3_result_cache	Nein	FALSE	Zwischenspeichert die SQL-Abfrage zum Entladen von Daten für die Amazon-S3 -Pfadzuordnung im Arbeitsspeicher, sodass dieselbe Abfrage nicht erneut in derselben Spark-Sitzung ausgeführt werden muss. Wird nur unterstützt wenn `autopushdown` aktiviert ist.
unload_s3_format	Nein	PARQUET	PARQUET – Entlädt die Abfrageergebnisse im Parquet-Format. TEXT – Entlädt die Abfrageergebnisse im durch Pipes getrennten Textformat.
sse_kms_key	Nein	N/A	Der AWS SSE-KMS-Schlüssel, der für die Verschlüsselung während des `UNL0AD` Betriebs verwendet werden soll, anstelle der Standardverschlüsselung für. AWS

Name	Erforderlich	Standard	Beschreibung
extracopyoptions	Nein	N/A	Eine Liste zusätzlicher Optionen, die beim Laden von Daten an den `COPY`-Befehl von Amazon Redshift angehängt werden, wie z. B. `TRUNCATECOLUMNS` oder `MAXERROR n` (weitere Optionen finden Sie unter COPY: Optionale Parameter). Beachten Sie, dass nur Optionen verwendet werden können, die am Ende des Befehls sinnvoll sind, da diese Optionen an das Ende des `COPY`-Befehls angefügt werden. Damit sollten die meisten Anwendungsfälle abgedeckt werden.

Name	Erforderlich	Standard	Beschreibung
csvnullstring (experimentell)	Nein	NULL	Der Zeichenfolgenwert, der für Nullen geschrieben werden soll, wenn Sie die CSV `tempformat` verwenden. Dies sollte ein Wert sein, der in Ihren tatsächlichen Daten nicht vorkommt.

Diese neuen Parameter können auf folgende Weise verwendet werden.

Neue Optionen zur Leistungsverbesserung

Der neue Konnektor bietet einige neue Optionen zur Leistungsverbesserung:

- `autopushdown`: Standardmäßig aktiviert.
- `autopushdown.s3_result_cache`: Standardmäßig deaktiviert.
- `unload_s3_format`: Standardmäßig `PARQUET`.

Informationen zur Verwendung dieser Optionen finden Sie unter Amazon-Redshift-Integration für Apache Spark. Es wird empfohlen, das `autopushdown.s3_result_cache` nicht einzuschalten, wenn Sie Lese- und Schreibvorgänge miteinander kombinieren, da die zwischengespeicherten Ergebnisse möglicherweise veraltete Informationen enthalten. Die Option `unload_s3_format` ist standardmäßig auf `PARQUET` für den `UNLOAD`-Befehl festgelegt, um die Leistung zu verbessern und die Speicherkosten zu reduzieren. Um das Standardverhalten des `UNLOAD`-Befehls zu verwenden, setzen Sie die Option auf `TEXT` zurück.

Neue Verschlüsselungsoption für das Lesen

Standardmäßig werden die Daten im temporären Ordner, den AWS Glue beim Lesen von Daten aus der Amazon-Redshift-Tabelle verwendet, mit der `SSE-S3`-Verschlüsselung verschlüsselt. Um Ihre Daten mit vom Kunden verwalteten Schlüsseln von AWS Key Management Service (AWS KMS) zu verschlüsseln, können Sie festlegen, dass KSMKey die Schlüssel-ID ist AWS KMS, und nicht die alte

Einstellungsoption (`"sse_kms_key" # kmsKey`) in Version 3.0. (`"extraunloadoptions" # s"ENCRYPTED KMS_KEY_ID '$kmsKey'"`) AWS Glue

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
  database = "database-name",
  table_name = "table-name",
  redshift_tmp_dir = args["TempDir"],
  additional_options = {"sse_kms_key":"<KMS_KEY_ID>"},
  transformation_ctx = "datasource0"
)
```

Unterstützung von IAM-basierter JDBC-URL

Der neue Connector unterstützt eine IAM-basierte JDBC-URL, sodass Sie keine geheime URL eingeben müssen. user/password Bei einer IAM-basierten JDBC-URL verwendet der Konnektor die Laufzeitrolle des Auftrags für den Zugriff auf die Amazon-Redshift-Datenquelle.

Schritt 1: Fügen Sie die folgende minimal erforderliche Richtlinie an Ihre AWS Glue-Auftrag-Laufzeitrolle an.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
                "arn:aws:redshift:<region>:<account>:dbgroup:<cluster name>/*",
                "arn:aws:redshift:*:<account>:dbuser:*/*",
                "arn:aws:redshift:<region>:<account>:dbname:<cluster name>/
<database name>"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "redshift:DescribeClusters",
            "Resource": "*"
```

```
        }
    ]
}
```

Schritt 2: Verwenden Sie die IAM-basierte JDBC-URL wie folgt. Geben Sie eine neue Option `DbUser` mit dem Amazon-Redshift-Benutzernamen an, mit dem Sie eine Verbindung herstellen.

```
conn_options = {
     // IAM-based JDBC URL
    "url": "jdbc:redshift:iam://<cluster name>:<region>/<database name>",
    "dbtable": dbtable,
    "redshiftTmpDir": redshiftTmpDir,
    "aws_iam_role": aws_iam_role,
    "DbUser": "<Redshift User name>" // required for IAM-based JDBC URL
    }

redshift_write = glueContext.write_dynamic_frame.from_options(
    frame=dyf,
    connection_type="redshift",
    connection_options=conn_options
)

redshift_read = glueContext.create_dynamic_frame.from_options(
    connection_type="redshift",
    connection_options=conn_options
)
```

> **Note**
>
> Ein `DynamicFrame` unterstützt derzeit nur eine IAM-basierte JDBC-URL mit einem `DbUser` im `GlueContext.create_dynamic_frame.from_options`-Workflow.

Migrieren von der AWS Glue-Version 3.0 zu Version 4.0

In AWS Glue 4.0 haben ETL-Jobs Zugriff auf einen neuen Amazon Redshift Spark-Konnektor und einen neuen JDBC-Treiber mit unterschiedlichen Optionen und Konfigurationen. Der neue Amazon-Redshift-Konnektor und -Treiber sind auf Leistung ausgelegt und gewährleisten die Transaktionskonsistenz Ihrer Daten. Diese Produkte sind in der Amazon-Redshift-Dokumentation dokumentiert. Weitere Informationen finden Sie unter:

- Amazon-Redshift-Integration für Apache Spark
- Amazon-Redshift-JDBC-Treiber, Version 2.1

Einschränkung für Tabellen-/Spaltennamen und Kennungen

Für den neuen Amazon-Redshift-Spark-Konnektor und -Treiber gelten strengere Anforderungen für den Redshift-Tabellennamen. Weitere Informationen finden Sie unter Namen und Kennungen zum Definieren des Namens Ihrer Amazon-Redshift-Tabelle. Der Workflow für Auftragslesezeichen funktioniert möglicherweise nicht mit einem Tabellennamen, der nicht mit den Regeln übereinstimmt, und mit bestimmten Zeichen, z. B. einem Leerzeichen.

Wenn Sie über veraltete Tabellen mit Namen verfügen, die nicht den Regeln für Namen und Kennungen entsprechen, und Probleme mit Lesezeichen sehen (Aufträge, die alte Amazon-Redshift-Tabellendaten neu verarbeiten), empfehlen wir Ihnen, Ihre Tabellennamen umzubenennen. Weitere Informationen finden Sie unter ALTER-TABLE-Beispiele.

Änderung des Standard-Tempformats in Dataframe

Der Spark-Konnektor der AWS Glue-Version 3.0 setzt `tempformat` beim Schreiben in Amazon Redshift standardmäßig auf CSV. Um konsistent zu sein, wird in der AWS Glue-Version 3.0 der `DynamicFrame` immer noch standardmäßig von dem `tempformat` verwendet, um `CSV` zu verwenden. Wenn Sie Spark Dataframe zuvor APIs direkt mit dem Amazon Redshift Spark-Konnektor verwendet haben, können Sie den in den `DataframeReader` Optionen/explizit `tempformat` auf CSV setzen. `Writer` Andernfalls wird `tempformat` im neuen Spark-Konnektor standardmäßig auf `AVRO` festgelegt.

Verhaltensänderung: Amazon-Redshift-Datentyp REAL dem Spark-Datentyp FLOAT anstelle von DOUBLE zuordnen

In der AWS Glue-Version 3.0 wird Amazon Redshift `REAL` in einen `DOUBLE`-Typ von Spark konvertiert. Der neue Amazon-Redshift-Spark-Konnektor hat das Verhalten so aktualisiert, dass der `REAL`-Typ von Amazon Redshift in den `FLOAT`-Typ von Spark konvertiert wird und umgekehrt. Wenn Sie einen älteren Anwendungsfall haben, bei dem Sie den `REAL`-Typ von Amazon Redshift weiterhin einem `DOUBLE`-Typ von Spark zuordnen möchten, können Sie die folgende Problemumgehung verwenden:

- Ordnen Sie für eine `DynamicFrame` den `Float`-Typ einem `Double`-Typ mit `DynamicFrame.ApplyMapping` zu. Für eine `Dataframe` müssen Sie `cast` verwenden.

Codebeispiel:

```
dyf_cast = dyf.apply_mapping([('a', 'long', 'a', 'long'), ('b', 'float', 'b',
 'double')])
```

Umgang mit dem VARBYTE-Datentyp

Bei der Arbeit mit den Datentypen AWS Glue 3.0 und Amazon Redshift konvertiert AWS Glue 3.0 Amazon Redshift in `VARBYTE` den `STRING` Spark-Typ. Der neueste Amazon Redshift Spark-Connector unterstützt den `VARBYTE` Datentyp jedoch nicht. Um diese Einschränkung zu umgehen, können Sie eine Redshift-Ansicht erstellen, die `VARBYTE` Spalten in einen unterstützten Datentyp umwandelt. Verwenden Sie dann den neuen Konnektor, um Daten aus dieser Ansicht statt aus der Originaltabelle zu laden. Dadurch wird die Kompatibilität gewährleistet und gleichzeitig der Zugriff auf Ihre `VARBYTE` Daten gewahrt.

Beispiel für eine Redshift-Abfrage:

```
CREATE VIEW view_name AS SELECT FROM_VARBYTE(varbyte_column, 'hex') FROM table_name
```

Kafka-Verbindungen

Sie können eine Kafka-Verbindung verwenden, um Kafka-Datenströme zu lesen und in sie zu schreiben, indem Sie Informationen verwenden, die in einer Datenkatalogtabelle gespeichert sind, oder indem Sie Informationen für den direkten Zugriff auf den Datenstrom bereitstellen. Die Verbindung unterstützt einen Kafka-Cluster oder einen Amazon Managed Streaming for Apache Kafka Kafka-Cluster. Sie können Informationen aus Kafka in einen Spark einlesen DataFrame und sie dann in einen AWS Glue DynamicFrame umwandeln. Sie können in einem JSON-Format in Kafka schreiben DynamicFrames . Wenn Sie direkt auf den Datenstrom zugreifen, verwenden Sie diese Optionen, um Informationen zum Zugriff auf den Datenstrom bereitzustellen.

Wenn Sie Datensätze aus einer Kafka-Streaming-Quelle verwenden `getCatalogSource` oder `create_data_frame_from_catalog` `getCatalogSink` oder `write_dynamic_frame_from_catalog` um Datensätze in Kafka zu schreiben, und der Job über die Datenkatalogdatenbank und die Tabellennameninformationen verfügt und diese verwenden kann, um einige grundlegende Parameter für das Lesen aus der Kafka-Streaming-Quelle abzurufen. Wenn Sie`getSource`,,`getCatalogSink`, `createDataFrameFromOptions` oder `getSourceWithFormatgetSinkWithFormat`, oder verwenden`create_data_frame_from_options`, müssen Sie diese

grundlegenden Parameter mithilfe der hier beschriebenen Verbindungsoptionen angeben. `write_dynamic_frame_from_catalog`

Sie können die Verbindungsoptionen für Kafka mithilfe der folgenden Argumente für die angegebenen Methoden in der `GlueContext` Klasse angeben.

- Scala
  - `connectionOptions`: mit `getSource`, `createDataFrameFromOptions`, `getSink` verwenden
  - `additionalOptions`: mit `getCatalogSource`, `getCatalogSink` verwenden
  - `options`: mit `getSourceWithFormat`, `getSinkWithFormat` verwenden
- Python
  - `connection_options`: mit `create_data_frame_from_options`, `write_dynamic_frame_from_options` verwenden
  - `additional_options`: mit `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog` verwenden
  - `options`: mit `getSource`, `getSink` verwenden

Hinweise und Einschränkungen zum Streaming von ETL-Aufträgen finden Sie unter the section called “Hinweise zu und Einschränkungen für Streaming-ETL”.

Kafka konfigurieren

Es gibt keine AWS Voraussetzungen, um eine Verbindung zu Kafka-Streams herzustellen, die über das Internet verfügbar sind.

Sie können eine AWS Glue Kafka-Verbindung erstellen, um Ihre Verbindungsdaten zu verwalten. Weitere Informationen finden Sie unter the section called “Erstellen einer Verbindung für einen Kafka-Datenstrom”. Geben Sie in Ihrer AWS Glue-Job-Konfiguration *`connectionName`* als Zusätzliche Netzwerkverbindung an und geben Sie dann in Ihrem Methodenaufruf den `connectionName` Parameter *`connectionName`* an.

In bestimmten Fällen müssen Sie zusätzliche Voraussetzungen konfigurieren:

- Wenn Sie Amazon Managed Streaming für Apache Kafka mit IAM-Authentifizierung verwenden, benötigen Sie eine entsprechende IAM-Konfiguration.

- Wenn Sie Amazon Managed Streaming für Apache Kafka innerhalb einer Amazon VPC verwenden, benötigen Sie eine entsprechende Amazon-VPC-Konfiguration. Sie müssen eine AWS Glue-Verbindung erstellen, die Amazon VPC-Verbindungsinformationen bereitstellt. Sie benötigen Ihre Jobkonfiguration, um die AWS Glue-Verbindung als zusätzliche Netzwerkverbindung einzubeziehen.

Weitere Informationen zu den Voraussetzungen für Streaming-ETL-Aufträgen finden Sie unter the section called “Streaming-ETL-Aufträge”.

Beispiel: Aus Kafka-Streams lesen

Verwendet in Verbindung mit the section called “forEachBatch”.

Beispiel für die Kafka-Streaming-Quelle:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Beispiel: In Kafka-Streams schreiben

Beispiele für das Schreiben an Kafka:

Beispiel mit der `getSink` Methode:

```
data_frame_datasource0 =
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
```

```
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Beispiel mit der `write_dynamic_frame.from_options` Methode:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Referenz zur Kafka-Verbindungsoption

Verwenden Sie beim Lesen die folgenden Verbindungsoptionen mit`"connectionType": "kafka"`:

- `"bootstrap.servers"`(Erforderlich) Eine Liste von URLs Bootstrap-Servern, z. B. als`b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Diese Option muss im API-Aufruf angegeben oder in den Tabellenmetadaten im Data Catalog definiert werden.
- `"security.protocol"` (Erforderlich) Das Protokoll, das für die Kommunikation mit Brokern verwendet wird. Die möglichen Werte sind `"SSL"` oder `"PLAINTEXT"`.
- `"topicName"` (Erforderlich) Eine durch Kommas getrennte Liste von Themen, die abonniert werden sollen. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.
- `"assign"`: (Erforderlich) Eine JSON-Zeichenfolge, welche den spezifischen `TopicPartitions` zum Konsumieren angeben. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.

  Beispiel: '{"topicA":[0,1],"topicB":[2,4]}'
- `"subscribePattern"`: (Erforderlich) Eine Java-Regex-Zeichenfolge, die die Themenliste identifiziert, die abonniert werden soll. Sie müssen nur eines der folgenden `"topicName"`, `"assign"` oder `"subscribePattern"` angeben.

  Beispiel: 'topic.*'
- `"classification"` (Erforderlich) Das von den Daten im Datensatz verwendete Dateiformat. Erforderlich, sofern nicht in Data Catalog angegeben.

- "delimiter" (Optional) Das verwendete Werttrennzeichen, wenn classification CSV ist. Der Standardwert ist „,„.
- "startingOffsets": (Optional) Die Ausgangsposition im Kafka-Thema, aus dem Daten gelesen werden sollen. Die möglichen Werte sind "earliest" oder "latest". Der Standardwert ist "latest".
- "startingTimestamp": (Optional, nur für AWS Glue Version 4.0 oder höher unterstützt) Der Zeitstempel des Datensatzes im Kafka-Thema, aus dem Daten gelesen werden sollen. Der mögliche Wert ist eine Zeitstempelzeichenfolge im UTC-Format im Muster yyyy-mm-ddTHH:MM:SSZ (wobei Z einen UTC-Zeitzonenversatz mit einem +/- darstellt. Beispiel: „2023-04-04T08:00:00-04:00“).

  Hinweis: In der Liste der Verbindungsoptionen des AWS Glue-Streaming-Skripts kann nur eine von 'StartingOffsets' oder 'StartingTimeStamp' vorhanden sein. Wenn Sie diese beiden Eigenschaften angeben, schlägt der Job fehl.
- "endingOffsets": (Optional) Der Endpunkt, wenn eine Batchabfrage beendet wird. Die möglichen Werte sind entweder "latest" oder eine JSON-Zeichenfolge, die einen Offset für das Ende jeder TopicPartition angibt.

  Für die JSON-Zeichenfolge lautet das Format {"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}. Der Wert -1 als Offset steht für "latest".
- "pollTimeoutMs": (Optional) Das Timeout in Millisekunden, um Daten von Kafka in Spark-Auftragsausführungen abzufragen. Der Standardwert ist 600000.
- "numRetries": (Optional) Die Anzahl, wie oft erneute Versuche durchgeführt werden sollen, bevor Kafka-Offsets nicht abgerufen werden. Der Standardwert ist 3.
- "retryIntervalMs": (Optional) Die Wartezeit in Millisekunden, bevor Sie erneut versuchen, Kafka-Offsets abzurufen. Der Standardwert ist 10.
- "maxOffsetsPerTrigger": (Optional) Die Ratengrenze für die maximale Anzahl von Offsets, die pro Triggerintervall verarbeitet werden. Die angegebene Gesamtzahl der Offsets wird proportional auf topicPartitionsvon verschiedenen Volumes aufgeteilt. Der Standardwert ist null, was bedeutet, dass der Verbraucher alle Offsets bis zum bekannten letzten Offset liest.
- "minPartitions": (Optional) Die gewünschte Mindestanzahl an Partitionen, die von Kafka gelesen werden sollen. Der Standardwert ist null, was bedeutet, dass die Anzahl der Spark-Partitionen gleich der Anzahl der Kafka-Partitionen ist.
- "includeHeaders": (Optional) Gibt an, ob die Kafka-Header eingeschlossen werden sollen. Wenn die Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem

Namen „glue_streaming_kafka_headers“ mit dem Typ `Array[Struct(key: String, value: String)]`. Der Standardwert ist „false“. Diese Option ist verfügbar in AWS Glue Version 3.0 oder höher.

- `"schema"`: (Erforderlich, wenn inferSchema auf „false“ festgelegt ist) Das Schema, das zur Verarbeitung der Nutzlast verwendet werden soll. Wenn die Klassifizierung `avro` ist, muss das bereitgestellte Schema im Avro-Schemaformat vorliegen. Wenn die Klassifizierung nicht `avro` ist, muss das bereitgestellte Schema im DDL-Schemaformat vorliegen.

  Im Folgenden finden Sie Beispiele für Schemata.

  Example in DDL schema format

  ```
  'column1' INT, 'column2' STRING , 'column3' FLOAT
  ```

  Example in Avro schema format

  ```
  {
  "type":"array",
  "items":
  {
  "type":"record",
  "name":"test",
  "fields":
  [
    {
      "name":"_id",
      "type":"string"
    },
    {
      "name":"index",
      "type":
      [
        "int",
        "string",
        "float"
      ]
    }
  ]
  }
  }
  ```

- `"inferSchema"`: (Optional) Der Standardwert ist „false“. Wenn auf „true“ gesetzt, wird das Schema zur Laufzeit von der Nutzlast in `foreachbatch` erkannt.
- `"avroSchema"`: (Veraltet) Parameter, der verwendet wird, um ein Schema von Avro-Daten anzugeben, wenn das Avro-Format verwendet wird. Dieser Parameter ist jetzt veraltet. Verwenden Sie den Parameter `schema`.
- `"addRecordTimestamp"`: (Optional) Wenn diese Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die den Zeitpunkt angibt, zu dem der entsprechende Datensatz beim Thema eingegangen ist. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.
- `"emitConsumerLagMetrics"`: (Optional) Wenn die Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, den das Thema empfangen hat, und dem Zeitpunkt, zu dem er eingeht, ausgegeben AWS Glue zu CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird unterstützt in AWS Glue Version 4.0 oder höher.

Verwenden Sie beim Schreiben die folgenden Verbindungsoptionen mit`"connectionType": "kafka"`:

- `"connectionName"`(Erforderlich) Name der AWS Glue-Verbindung, die für die Verbindung mit dem Kafka-Cluster verwendet wird (ähnlich der Kafka-Quelle).
- `"topic"`(Erforderlich) Wenn eine Themenspalte existiert, wird ihr Wert als Thema verwendet, wenn die angegebene Zeile in Kafka geschrieben wird, sofern die Themenkonfigurationsoption nicht festgelegt ist. Das heißt, die `topic` Konfigurationsoption überschreibt die Themenspalte.
- `"partition"`(Optional) Wenn eine gültige Partitionsnummer angegeben ist, `partition` wird diese beim Senden des Datensatzes verwendet.

  Wenn keine Partition angegeben ist, aber eine vorhanden `key` ist, wird eine Partition anhand eines Hashs des Schlüssels ausgewählt.

  Falls `key` weder noch vorhanden `partition` ist, wird eine Partition auf der Grundlage von Sticky-Partitionierung ausgewählt. Diese Änderungen werden erst dann vorgenommen, wenn für die Partition mindestens batch.size-Byte erzeugt werden.
- `"key"`(Optional) Wird für die Partitionierung verwendet, wenn der Wert Null ist. `partition`
- `"classification"`(Optional) Das von den Daten im Datensatz verwendete Dateiformat. Wir unterstützen nur JSON, CSV und Avro.

Mit dem Avro-Format können wir ein benutzerdefiniertes AvroSchema für die Serialisierung bereitstellen. Beachten Sie jedoch, dass dieses auch in der Quelle für die Deserialisierung bereitgestellt werden muss. Andernfalls verwendet es standardmäßig den Apache für die Serialisierung. AvroSchema

Darüber hinaus können Sie die Kafka-Senke nach Bedarf feinabstimmen, indem Sie die Konfigurationsparameter des Kafka-Producers aktualisieren. Beachten Sie, dass es keine Zulassungsliste für Verbindungsoptionen gibt, da alle Schlüssel-Wert-Paare unverändert auf der Senke gespeichert werden.

Es gibt jedoch eine kleine Liste von Optionen, die nicht wirksam werden. Weitere Informationen finden Sie unter Kafka-spezifische Konfigurationen.

Azure-Cosmos-DB-Verbindungen

Sie können AWS Glue for Spark verwenden, um mithilfe der NoSQL-API in AWS Glue 4.0 und späteren Versionen aus vorhandenen Containern in Azure Cosmos DB zu lesen und in diese zu schreiben. Sie können definieren, was mit einer SQL-Abfrage aus Azure Cosmos DB gelesen werden soll. Sie stellen mithilfe eines Azure Cosmos DB-Schlüssels, der über eine AWS Glue-Verbindung gespeichert ist, eine Verbindung zu Azure Cosmos DB her. AWS Secrets Manager

Weitere Informationen zu Azure Cosmos DB for NoSQL finden Sie in der Azure-Dokumentation.

Konfigurieren von Azure-Cosmos DB-Verbindungen

Um von AWS Glue aus eine Verbindung mit Azure Cosmos DB herzustellen, müssen Sie Ihren Azure Cosmos DB-Schlüssel erstellen und in einem AWS Secrets Manager Geheimnis speichern und dieses Geheimnis dann einer Azure Cosmos DB AWS Glue-Verbindung zuordnen.

Voraussetzungen:

- In Azure müssen Sie einen Azure Cosmos DB-Schlüssel für die Verwendung durch AWS Glue identifizieren oder generieren. `cosmosKey` Weitere Informationen finden Sie unter Sicherer Zugriff auf Daten in Azure Cosmos DB in der Azure-Dokumentation.

Eine Verbindung zu Azure Cosmos DB konfigurieren Sie wie folgt:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihres Azure Cosmos DB-Schlüssels ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das

in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `spark.cosmos.accountKey` mit dem Wert. *cosmosKey*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für die future Verwendung in AWS Glue bei. *connectionName*

   - Wählen Sie als Verbindungstyp Azure Cosmos DB aus.
   - Wenn Sie ein AWS Geheimnis auswählen, geben Sie an*secretName*.

Nachdem Sie eine AWS Glue Azure Cosmos DB-Verbindung erstellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue-Job ausführen:

- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *secretName* Leserechte.
- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen aus Container in Azure Cosmos DB für NoSQL

Voraussetzungen:

- Ein Container in Azure Cosmos DB für NoSQL, aus dem Sie lesen möchten. Sie benötigen Identifikationsinformationen für den Container.

  Ein Container in Azure Cosmos DB für NoSQL wird anhand seiner Datenbank und seines Containers identifiziert. Sie müssen die Datenbank- und Container-Namen angeben*cosmosContainerName*, wenn Sie eine Verbindung zur Azure Cosmos for NoSQL-API herstellen. *cosmosDBName*
- Eine AWS Glue Azure Cosmos DB-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen bereitstellt. Führen Sie dazu die Schritte im vorherigen Verfahren zum Konfigurieren einer Verbindung zu Azure Cosmos DB aus. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
azurecosmos_read = glueContext.create_dynamic_frame.from_options(
    connection_type="azurecosmos",
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName,
    }
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Ihre zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `query` konfigurieren.

Zum Beispiel:

```
azurecosmos_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="azurecosmos",
    connection_options={
        "connectionName": "connectionName",
        "spark.cosmos.database": cosmosDBName,
        "spark.cosmos.container": cosmosContainerName,
        "spark.cosmos.read.customQuery": "query"
    }
)
```

Schreiben in Container in Azure Cosmos DB für NoSQL

In diesem Beispiel werden Informationen aus einer vorhandenen DynamicFrame Datenbank *dynamicFrame* in Azure Cosmos DB geschrieben. Wenn der Container bereits Informationen enthält, hängt AWS Glue Daten von Ihrem DynamicFrame an. Wenn die Informationen im Container ein anderes Schema haben als die Informationen, die Sie schreiben, treten Fehler auf.

Voraussetzungen:

- Eine Azure-Cosmos-DB-Tabelle, in die Sie schreiben möchten. Sie benötigen Identifikationsinformationen für den Container. Sie müssen den Container erstellen, bevor Sie die Verbindungsmethode aufrufen.

  Ein Container in Azure Cosmos DB für NoSQL wird anhand seiner Datenbank und seines Containers identifiziert. Sie müssen die Datenbank- und Container-Namen

angeben*cosmosContainerName*, wenn Sie eine Verbindung zur Azure Cosmos for NoSQL-API herstellen. *cosmosDBName*

- Eine AWS Glue Azure Cosmos DB-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen bereitstellt. Führen Sie dazu die Schritte im vorherigen Verfahren zum Konfigurieren einer Verbindung zu Azure Cosmos DB aus. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
azurecosmos_write = glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="azurecosmos",
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName
)
```

Referenz zur Azure-Cosmos-DB-Verbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue Azure Cosmos DB-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkstandortinformationen für Ihre Verbindungsmethode bereitstellt.
- `spark.cosmos.database` – Erforderlich. Wird für Lesen/Schreiben verwendet. Zulässige Werte: Datenbanknamen. Name der Azure Cosmos DB für NoSQL-Datenbank.
- `spark.cosmos.container` – Erforderlich. Wird für Lesen/Schreiben verwendet. Zulässige Werte: Containernamen. Name des Containers in Azure Cosmos DB für NoSQL.
- `spark.cosmos.read.customQuery` – Wird zum Lesen verwendet. Gültige Werte: SELECT-SQL-Abfragen. Benutzerdefinierte Abfrage zur Auswahl von Dokumenten, die gelesen werden sollen.

Azure-SQL-Verbindungen

Sie können AWS Glue for Spark verwenden, um aus Tabellen auf verwalteten Azure SQL-Instanzen in AWS Glue 4.0 und späteren Versionen zu lesen und in Tabellen zu schreiben. Sie können definieren, was mit einer SQL-Abfrage aus Azure SQL gelesen werden soll. Sie stellen mithilfe von

Benutzer- und Kennwortanmeldeinformationen, die AWS Secrets Manager über eine AWS Glue-Verbindung gespeichert sind, eine Verbindung zu Azure SQL her.

Weitere Informationen zu Azure SQL finden Sie in der Azure-SQL-Dokumentation.

Konfigurieren von Azure-SQL-Verbindungen

Um von AWS Glue aus eine Verbindung zu Azure SQL herzustellen, müssen Sie Ihre Azure SQL-Anmeldeinformationen erstellen und in einem AWS Secrets Manager geheimen Ordner speichern und dieses Geheimnis dann einer Azure SQL AWS Glue-Verbindung zuordnen.

Eine Verbindung zu Azure SQL konfigurieren Sie wie folgt:

1. Erstellen Sie unter Verwendung Ihrer Azure SQL-Anmeldeinformationen ein Geheimnis. AWS Secrets Manager Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *azuresqlUsername*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *azuresqlPassword*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für die future Verwendung in AWS Glue bei. *connectionName*

   - Wählen Sie als Verbindungstyp Azure SQL aus.
   - Wenn Sie eine Azure-SQL-URL angeben, geben Sie eine JDBC-Endpunkt-URL an.

     Die URL muss das folgende Format aufweisen:
     `jdbc:sqlserver://`*databaseServerName*`:`*databasePort*`;databaseName=`*azuresqlDBnam*

     AWS Glue benötigt die folgenden URL-Eigenschaften:

     - `databaseName` – Eine Standarddatenbank in Azure SQL, mit der eine Verbindung hergestellt werden kann.

Weitere Informationen zu JDBC URLs für Azure SQL Managed Instances finden Sie in der Microsoft-Dokumentation.

- Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an: *secretName*

Nachdem Sie eine AWS Glue Azure SQL-Verbindung erstellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue-Job ausführen:

- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *secretName* Leserechte.
- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen in Azure-SQL-Tabellen

Voraussetzungen:

- Eine Azure-SQL-Tabelle, aus der gelesen werden soll. Sie benötigen Identifikationsinformationen für die Tabelle *databaseName* und*tableIdentifier*.

  Eine Azure-SQL-Tabelle wird anhand ihrer Datenbank, ihres Schemas und ihres Tabellennamens identifiziert. Sie müssen den Datenbanknamen und den Tabellennamen angeben, wenn Sie eine Verbindung zu Azure SQL herstellen. Sie müssen auch das Schema angeben, falls es sich nicht um das Standardschema „public“ handelt. Die Datenbank wird über eine URL-Eigenschaft in*connectionName*, ein Schema und einen Tabellennamen über die bereitgestelltdbtable.
- Eine AWS Glue Azure SQL-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu Azure SQL zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
azuresql_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
```

```
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Ihre zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `query` konfigurieren.

Zum Beispiel:

```
azuresql_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Schreiben in Azure-SQL-Tabellen

In diesem Beispiel werden Informationen aus einer vorhandenen DynamicFrame Datei *`dynamicFrame`* in Azure SQL geschrieben. Wenn die Tabelle bereits Informationen enthält, hängt AWS Glue Daten von Ihnen DynamicFrame an.

Voraussetzungen:

- Eine Azure-SQL-Tabelle, in die Sie schreiben möchten. Sie benötigen Identifikationsinformationen für die Tabelle, *`databaseName`* und*`tableIdentifier`*.

  Eine Azure-SQL-Tabelle wird anhand ihrer Datenbank, ihres Schemas und ihres Tabellennamens identifiziert. Sie müssen den Datenbanknamen und den Tabellennamen angeben, wenn Sie eine Verbindung zu Azure SQL herstellen. Sie müssen auch das Schema angeben, falls es sich nicht um das Standardschema „public“ handelt. Die Datenbank wird über eine URL-Eigenschaft in*`connectionName`*, ein Schema und einen Tabellennamen über die bereitgestelltdbtable`.
- Azure-SQL-Authentifizierungsinformationen. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu Azure SQL zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*`connectionName`*.

Zum Beispiel:

```
azuresql_write = glueContext.write_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
```

```
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
)
```

Referenz zur Azure-SQL-Verbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue Azure SQL-Verbindung, die so konfiguriert ist, dass sie Authentifizierungsinformationen für Ihre Verbindungsmethode bereitstellt.
- `databaseName` – Wird zum Lesen/Schreiben verwendet. Zulässige Werte: Azure-SQL-Datenbanknamen. Der Name der Datenbank in Azure SQL, mit der eine Verbindung hergestellt werden soll.
- `dbtable` – Zum Schreiben erforderlich, zum Lesen erforderlich, sofern `query` nicht angegeben ist. Wird für Read/Write. Valid Values: Names of Azure SQL tables, or period separated schema/table Namenskombinationen verwendet. Wird verwendet, um die Tabelle und das Schema anzugeben, die die Tabelle identifizieren, zu der eine Verbindung hergestellt werden soll. Das Standardschema ist „public“. Wenn sich Ihre Tabelle nicht in einem Standardschema befindet, geben Sie diese Informationen in das Formular *`schemaName`*`.`*`tableName`* ein.
- `query` – Wird zum Lesen verwendet. Eine Transact-SQL-SELECT-Abfrage, die definiert, was beim Lesen aus Azure SQL abgerufen werden soll. Weitere Informationen finden Sie in der Microsoft-Dokumentation.

BigQuery Verbindungen

In AWS Glue 4.0 und späteren Versionen können Sie Glue for Spark verwenden, um aus Tabellen BigQuery in AWS Google zu lesen und in Tabellen zu schreiben. Sie können BigQuery mit einer Google-SQL-Abfrage aus lesen. Sie stellen eine Verbindung BigQuery mit Anmeldeinformationen her, AWS Secrets Manager die über eine AWS Glue-Verbindung gespeichert sind.

Weitere Informationen zu Google BigQuery finden Sie auf der Google BigQuery Cloud-Website.

BigQuery Verbindungen konfigurieren

Um BigQuery von AWS Glue aus eine Verbindung zu Google herzustellen, müssen Sie Ihre Anmeldeinformationen für die Google Cloud Platform erstellen und in einem AWS Secrets Manager geheimen Ordner speichern und dieses Geheimnis dann mit einer Google BigQuery AWS Glue-Verbindung verknüpfen.

So konfigurieren Sie eine Verbindung zu BigQuery:

1. Erstellen und identifizieren Sie in der Google Cloud Platform relevante Ressourcen:

   - Erstellen oder identifizieren Sie ein GCP-Projekt mit BigQuery Tabellen, zu denen Sie eine Verbindung herstellen möchten.
   - Aktivieren Sie die BigQuery API. Weitere Informationen finden Sie unter Verwenden der BigQuery Storage Read API zum Lesen von Tabellendaten.

2. Erstellen und exportieren Sie in Google Cloud Platform Anmeldeinformationen für ein Dienstkonto:

   Sie können den Assistenten für BigQuery Anmeldeinformationen verwenden, um diesen Schritt zu beschleunigen: Anmeldeinformationen erstellen.

   Folgen Sie der Anleitung unter Dienstkonten erstellen, um ein Dienstkonto in GCP zu erstellen.

   - Wählen Sie bei der Projektauswahl das Projekt aus, das Ihre BigQuery Tabelle enthält.
   - Wenn Sie GCP-IAM-Rollen für Ihr Dienstkonto auswählen, fügen Sie eine Rolle hinzu oder erstellen Sie eine Rolle, die entsprechende Berechtigungen zum Ausführen von BigQuery Jobs zum Lesen, Schreiben oder Erstellen BigQuery von Tabellen gewährt.

   Folgen Sie der Anleitung unter Einen Dienstkontoschlüssel erstellen, um Anmeldeinformationen für Ihr Dienstkonto zu erstellen.

   - Wählen Sie für den Schlüsseltyp JSON aus.

   Sie sollten jetzt eine JSON-Datei mit Anmeldeinformationen für Ihr Dienstkonto heruntergeladen haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
```

```
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
}
```

3. base64-kodieren Sie die heruntergeladene Datei mit den Anmeldeinformationen. In einer AWS CloudShell Sitzung oder ähnlichem können Sie dies von der Befehlszeile aus tun, indem Sie den Befehl ausführen. `cat` *`credentialsFile.json`* `| base64 -w 0` Behalten Sie die Ausgabe dieses Befehls bei,*`credentialString`*.
4. Erstellen Sie AWS Secrets Manager unter mit Ihren Anmeldedaten für die Google Cloud Platform ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `credentials` mit dem Wert. *`credentialString`*
5. Stellen AWS Sie im Glue-Datenkatalog eine Verbindung her, indem Sie die Schritte unter befolgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *`connectionName`*
   - Wählen Sie bei der Auswahl eines Verbindungstyps Google aus BigQuery.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*`secretName`*.
6. Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *`secretName`* Leserechte.
7. Stellen *`connectionName`* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Aus BigQuery Tabellen lesen

Voraussetzungen:

- Eine BigQuery Tabelle, aus der Sie lesen möchten. Sie benötigen die BigQuery Tabellen- und Datensatznamen im Formular`[dataset].[table]`. Nennen wir das*`tableName`*.
- Das Abrechnungsprojekt für die BigQuery Tabelle. Sie benötigen den Namen des Projekts,*`parentProject`*. Wenn es kein übergeordnetes Fakturierungsprojekt gibt, verwenden Sie das Projekt, das die Tabelle enthält.

- BigQuery Informationen zur Authentifizierung. Gehen Sie wie folgt vor, um Ihre Verbindungsdaten mit AWS Glue zu verwalten, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
bigquery_read = glueContext.create_dynamic_frame.from_options(
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "sourceType": "table",
        "table": "tableName",
    }
```

Sie können auch eine Abfrage angeben, um die an Sie zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `query`, `sourceType`, `viewsEnabled` und `materializationDataset` konfigurieren.

Zum Beispiel:

Zusätzliche Voraussetzungen:

Sie müssen einen BigQuery Datensatz erstellen oder identifizieren*materializationDataset*, in den Sie materialisierte Ansichten für Ihre Abfragen schreiben BigQuery können.

Sie müssen Ihrem Dienstkonto die entsprechenden GCP-IAM-Berechtigungen gewähren, um darin Tabellen erstellen zu können. *materializationDataset*

```
glueContext.create_dynamic_frame.from_options(
            connection_type="bigquery",
            connection_options={
                "connectionName": "connectionName",
                "materializationDataset": materializationDataset,
                "parentProject": "parentProject",
                "viewsEnabled": "true",
                "sourceType": "query",
                "query": "select * from bqtest.test"
            }
        )
```

In Tabellen schreiben BigQuery

In diesem Beispiel wird direkt in den BigQuery Dienst geschrieben. BigQuery unterstützt auch die „indirekte“ Schreibmethode. Weitere Informationen zur Konfiguration indirekter Schreibvorgänge finden Sie unter the section called “Verwenden Sie indirektes Schreiben mit Google BigQuery”.

Voraussetzungen:

- Eine BigQuery Tabelle, in die Sie schreiben möchten. Sie benötigen die BigQuery Tabellen- und Datensatznamen im Formular`[dataset].[table]`. Sie können auch einen neuen Tabellennamen angeben, der automatisch erstellt wird. Nennen wir das*`tableName`*.
- Das Abrechnungsprojekt für die BigQuery Tabelle. Sie benötigen den Namen des Projekts,*`parentProject`*. Wenn es kein übergeordnetes Fakturierungsprojekt gibt, verwenden Sie das Projekt, das die Tabelle enthält.
- BigQuery Informationen zur Authentifizierung. Gehen Sie wie folgt vor, um Ihre Verbindungsdaten mit AWS Glue zu verwalten, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*`connectionName`*.

Zum Beispiel:

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "writeMethod": "direct",
        "table": "tableName",
    }
)
```

BigQuery Referenz zur Verbindungsoption

- `project` – Standard: Standard für das Google Cloud-Dienstkonto. Wird für Lesen/Schreiben verwendet. Der Name eines Google Cloud-Projekts, das mit Ihrer Tabelle verknüpft ist.
- `table` – (Erforderlich) Wird zum Lesen/Schreiben verwendet. Der Name Ihrer BigQuery Tabelle im Format`[[project:]dataset.]`.

- `dataset` – Erforderlich, wenn nicht über die Option `table` definiert. Wird für Lesen/Schreiben verwendet. Der Name des Datensatzes, der Ihre BigQuery Tabelle enthält.
- `parentProject` – Standard: Standard für das Google Cloud-Dienstkonto. Wird für Lesen/Schreiben verwendet. Der Name eines Google Cloud-Projekts, das mit `project` verknüpft ist und für die Fakturierung verwendet wird.
- `sourceType` – Wird zum Lesen verwendet. Beim Lesen erforderlich. Gültige Werte:`table`, `query` Informiert AWS Glue darüber, ob Sie nach Tabelle oder nach Abfrage lesen werden.
- `materializationDataset` – Wird zum Lesen verwendet. Gültige Werte: Zeichenfolgen. Der Name eines BigQuery Datensatzes, der zum Speichern von Materialisierungen für Ansichten verwendet wird.
- `viewsEnabled` – Wird zum Lesen verwendet. Standard: falsch. Zulässige Werte: true, false. Konfiguriert, ob Ansichten verwendet BigQuery werden.
- `query` – Wird zum Lesen verwendet. Wird verwendet, wenn `viewsEnabled` „true“ ist. Eine GoogleSQL-DQL-Abfrage.
- `temporaryGcsBucket` – Wird zum Schreiben verwendet. Erforderlich, wenn `writeMethod` auf Standard (`indirect`) eingestellt ist. Name eines Google Cloud Storage-Buckets, in dem beim Schreiben eine Zwischenform Ihrer Daten gespeichert wird. BigQuery
- `writeMethod` – Standardwert: `indirect`. Zulässige Werte: `direct`, `indirect`. Wird zum Schreiben verwendet. Gibt die Methode an, mit der Ihre Daten geschrieben werden.
  - Wenn auf gesetzt`direct`, schreibt Ihr Connector mithilfe der BigQuery Storage Write-API.
  - Wenn auf gesetzt`indirect`, schreibt Ihr Connector in Google Cloud Storage und überträgt ihn dann BigQuery mithilfe eines Ladevorgangs an Google Cloud Storage. Ihr Google Cloud-Dienstkonto benötigt entsprechende GCS-Berechtigungen.

Verwenden Sie indirektes Schreiben mit Google BigQuery

In diesem Beispiel wird indirektes Schreiben verwendet, bei dem Daten in Google Cloud Storage geschrieben und nach Google kopiert BigQuery werden.

Voraussetzungen:

Sie benötigen einen temporären Google Cloud Storage-Bucket,*`temporaryBucket`*.

Für den Zugriff auf die GCP-IAM-Rolle für das GCP-Dienstkonto von AWS Glue sind entsprechende GCS-Berechtigungen erforderlich. *`temporaryBucket`*

Zusätzliche Konfiguration:

Um indirektes Schreiben zu konfigurieren mit: BigQuery

1. Bewerten Sie the section called “Konfiguration BigQuery” und suchen oder laden Sie die JSON-Datei mit Ihren GCP-Anmeldeinformationen erneut herunter. Identifizieren Sie *secretName* das AWS Secrets Manager Geheimnis für die Google BigQuery AWS Glue-Verbindung, die in Ihrem Job verwendet wird.
2. Laden Sie die JSON-Datei mit Ihren Anmeldeinformationen an einem entsprechend sicheren Amazon-S3-Speicherort hoch. Behalten Sie den Pfad zur Datei *s3secretpath* für future Schritte bei.
3. Bearbeiten*secretName*, `spark.hadoop.google.cloud.auth.service.account.json.keyfile` Schlüssel hinzufügen. Legen Sie den Wert auf *s3secretpath* fest.
4. Erteilen Sie Ihrem AWS Glue-Job Amazon S3 *s3secretpath* IAM-Zugriffsberechtigungen.

Sie können jetzt den Speicherort Ihres temporären GCS-Bucket für Ihre Schreibmethode angeben. Sie müssen `writeMethod` nicht angeben, da `indirect` üblicherweise die Standardeinstellung ist.

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "temporaryGcsBucket": "temporaryBucket",
        "table": "tableName",
    }
)
```

JDBC-Verbindungen

Bestimmte, in der Regel relationale Datenbanktypen unterstützen die Verbindung über den JDBC-Standard. Weitere Informationen zu JDBC finden Sie in der Java JDBC API-Dokumentation. AWS Glue unterstützt nativ die Verbindung zu bestimmten Datenbanken über ihre JDBC-Konnektoren — die JDBC-Bibliotheken werden in Glue Spark-Jobs bereitgestellt. AWS Wenn Sie mithilfe von AWS -Glue-Bibliotheken eine Verbindung zu diesen Datenbanktypen herstellen, haben Sie Zugriff auf eine Reihe von Standardoptionen.

Zu den JDBC-connectionType-Werten gehören:

- `"connectionType": "sqlserver"`: Bezeichnet eine Verbindung zu einer Microsoft SQL Server-Datenbank.
- `"connectionType": "mysql"`: Bezeichnet eine Verbindung zu einer MySQL-Datenbank.
- `"connectionType": "oracle"`: Bezeichnet eine Verbindung zu einer Oracle-Datenbank.
- `"connectionType": "postgresql"`: Bezeichnet eine Verbindung zu einer PostgreSQL-Datenbank.
- `"connectionType": "redshift"`: Bezeichnet eine Verbindung zu einer Amazon-Redshift-Datenbank. Weitere Informationen finden Sie unter the section called “Redshift-Verbindungen”.

In der folgenden Tabelle sind die JDBC-Treiberversionen aufgeführt, die AWS Glue unterstützt.

Produkt	JDBC-Treiberversionen für Glue 5.0	JDBC-Treiberversionen für Glue 4.0	JDBC-Treiberversionen für Glue 3.0	JDBC-Treiberversionen für Glue 0.9, 1.0, 2.0
Microsoft SQL Server	10.2.0	9.4,0	7.x	6.x
MySQL	8,0,33	8.0.23	8.0.23	5.1
Oracle Database	23,3,0,23,09	21,7	21,1	11.2
PostgreSQL	42,7,3	42,3,6	42,2,18	42.1.x
Amazon Redshift *	Rotshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017

* Für den Amazon Redshift Redshift-Verbindungstyp werden alle anderen Optionsname/Wert-Paare, die in den Verbindungsoptionen für eine JDBC-Verbindung enthalten sind, einschließlich Formatierungsoptionen, direkt an das zugrunde liegende SparkSQL übergeben. DataSource In AWS Glue with Spark-Jobs in AWS Glue 4.0 und späteren Versionen verwendet der native AWS Glue-Connector für Amazon Redshift die Amazon Redshift Redshift-Integration für Apache Spark. Weitere Informationen finden Sie unter Amazon-Redshift-Integration für Apache Spark. In früheren Versionen finden Sie Informationen zur Amazon-Redshift-Datenquelle für Spark.

Informationen zum Konfigurieren Ihrer Amazon VPC für die Verbindung mit Amazon-RDS-Datenspeichern über JDBC finden Sie unter the section called “Amazon VPC für die Verbindung mit Amazon RDS-Datenspeichern einrichten”.

> **Note**
>
> AWS Glue-Jobs werden während eines Laufs nur einem Subnetz zugeordnet. Dies kann sich auf Ihre Fähigkeit auswirken, über denselben Auftrag eine Verbindung zu mehreren Datenquellen herzustellen. Dieses Verhalten ist nicht auf JDBC-Quellen beschränkt.

Themen

- Referenz zur JDBC-Verbindungsoption
- sampleQuery verwenden
- Benutzerdefinierten JDBC-Treiber verwenden
- Parallel aus JDBC-Tabellen lesen
- Einrichtung von Amazon VPC für JDBC-Verbindungen zu Amazon RDS-Datenspeichern von AWS Glue

Referenz zur JDBC-Verbindungsoption

Wenn Sie bereits eine JDBC AWS Glue-Verbindung definiert haben, können Sie die darin definierten Konfigurationseigenschaften wie URL, Benutzer und Passwort wiederverwenden. Sie müssen sie also nicht im Code als Verbindungsoptionen angeben. Diese Funktion ist in AWS Glue 3.0 und späteren Versionen verfügbar. Verwenden Sie dazu die folgenden Verbindungseigenschaften:

- `"useConnectionProperties"`: Setzen Sie den Wert auf „true“, um anzugeben, dass Sie die Konfiguration von einer Verbindung aus verwenden möchten.
- `"connectionName"`: Geben Sie den Verbindungsnamen ein, aus dem die Konfiguration abgerufen werden soll. Die Verbindung muss in derselben Region wie der Auftrag definiert sein.

Verwenden Sie diese Verbindungsoptionen mit JDBC-Verbindungen:

- `"url"`: (Erforderlich) Die JDBC-URL für die Datenbank.
- `"dbtable"`: (Erforderlich) Die Datenbanktabelle, aus der gelesen werden soll. Bei JDBC-Datenspeichern, die von Schemata innerhalb einer Datenbank unterstützen, geben Sie

`schema.table-name` an. Wenn kein Schema angegeben ist, wird der Standardwert "öffentliches" Schema verwendet.

- `"user"`: (Erforderlich) Der Benutzername, der beim herstellen der Verbindung verwendet werden soll.
- `"password"`: (Erforderlich) Das Passwort für die Verbindung.
- (Optional) Mit den folgenden Optionen können Sie einen benutzerdefinierten JDBC-Treiber bereitstellen. Verwenden Sie diese Optionen, wenn Sie einen Treiber verwenden müssen, den AWS Glue nicht nativ unterstützt.

  ETL-Aufträge können verschiedene JDBC-Treiberversionen für die Datenquelle und das Ziel verwenden, selbst wenn Quelle und Ziel dasselbe Datenbankprodukt sind. Auf diese Weise können Sie Daten zwischen Quell- und Zieldatenbanken mit unterschiedlichen Versionen migrieren. Um diese Optionen zu verwenden, müssen Sie zuerst die JAR-Datei des JDBC-Treibers in Amazon S3 hochladen.

  - `"customJdbcDriverS3Path"`: Der Amazon-S3-Pfad des benutzerdefinierten JDBC-Treibers.
  - `"customJdbcDriverClassName"`: Der Klassenname des JDBC-Treibers.
- `"bulkSize"`: (Optional) Wird verwendet, um parallele Inserts zu konfigurieren, um Massenladungen in JDBC-Ziele zu beschleunigen. Geben Sie einen ganzzahligen Wert für den Parallelitätsgrad an, der beim Schreiben oder Einfügen von Daten verwendet werden soll. Diese Option ist hilfreich, um die Leistung beim Schreiben in Datenbanken wie das Arch User Repository (AUR) zu verbessern.
- `"hashfield"` (Optional) Eine Zeichenfolge, die zur Angabe des Namens einer Spalte in der JDBC-Tabelle verwendet wird, die zum Aufteilen der Daten in Partitionen beim parallelen Lesen aus JDBC-Tabellen verwendet werden soll. Geben Sie „hashfield“ ODER „hashexpression“ an. Weitere Informationen finden Sie unter the section called “Parallel aus JDBC lesen”.
- `"hashexpression"` (Optional) Eine SQL-Auswahlklausel, die eine ganze Zahl zurückgibt. Wird verwendet, um die Daten in einer JDBC-Tabelle beim parallelen Lesen aus JDBC-Tabellen in Partitionen zu unterteilen. Geben Sie „hashfield“ ODER „hashexpression“ an. Weitere Informationen finden Sie unter the section called “Parallel aus JDBC lesen”.
- `"hashpartitions"` (Optional) Eine positive Ganzzahl. Wird verwendet, um die Anzahl der parallelen Lesevorgänge der JDBC-Tabelle anzugeben, wenn parallel aus JDBC-Tabellen gelesen wird. Standard: 7. Weitere Informationen finden Sie unter the section called “Parallel aus JDBC lesen”.
- `"sampleQuery"`: (Optional) Eine benutzerdefinierte SQL-Abfrageanweisung. Wird verwendet, um eine Teilmenge von Informationen in einer Tabelle anzugeben, um ein Beispiel des Tabelleninhalts

abzurufen. Wenn es ohne Rücksicht auf Ihre Daten konfiguriert wird, kann es weniger effizient sein als DynamicFrame Methoden, was zu Timeouts oder Speichermangel führen kann. Weitere Informationen finden Sie unter the section called “sampleQuery verwenden”.

- `"enablePartitioningForSampleQuery"`: (Optional) Ein boolescher Wert. Standard: falsch. Wird verwendet, um das parallele Lesen aus JDBC-Tabellen bei Angabe von `sampleQuery` zu ermöglichen. Wenn auf true gesetzt, **`sampleQuery`** muss es mit „where“ oder „and“ enden, damit AWS Glue Partitionierungsbedingungen anhängt. Weitere Informationen finden Sie unter the section called “sampleQuery verwenden”.
- `"sampleSize"`: (Optional) Eine positive Ganzzahl. Begrenzt die Anzahl der von der Beispielabfrage zurückgegebenen Zeilen. Funktioniert nur, wenn `enablePartitioningForSampleQuery` true ist. Wenn die Partitionierung nicht aktiviert ist, sollten Sie stattdessen `"limit x"` direkt in `sampleQuery` einfügen, um die Größe zu begrenzen. Weitere Informationen finden Sie unter the section called “sampleQuery verwenden”.

sampleQuery verwenden

Dieser Abschnitt erläutert die Verwendung von `sampleQuery`, `sampleSize` und `enablePartitioningForSampleQuery`.

`sampleQuery` kann ein effizienter Weg sein, um ein paar Zeilen Ihres Datensatzes abzufragen. In der Standardeinstellung wird die Abfrage von einem einzigen Ausführer ausgeführt. Wenn es ohne Rücksicht auf Ihre Daten konfiguriert wird, kann es weniger effizient sein als DynamicFrame Methoden, was zu Timeouts oder Fehlern aufgrund unzureichender Speicherkapazität führen kann. Das Ausführen von SQL in der zugrunde liegenden Datenbank als Teil Ihrer ETL-Pipeline ist im Allgemeinen nur aus Leistungsgründen erforderlich. Wenn Sie eine Vorschau einiger Zeilen Ihres Datensatzes anzeigen möchten, sollten Sie die Verwendung von the section called “show” in Betracht ziehen. Wenn Sie versuchen, Ihren Datensatz mit SQL zu transformieren, sollten Sie erwägen, eine SparkSQL-Transformation für Ihre Daten in einem DataFrame Formular the section called “toDF” zu definieren.

Während Ihre Abfrage eine Vielzahl von Tabellen manipulieren kann, ist `dbtable` weiterhin erforderlich.

Verwendung von sampleQuery zum Abrufen eines Beispiels Ihrer Tabelle

Wenn Sie das Standardverhalten von SampleQuery verwenden, um eine Stichprobe Ihrer Daten abzurufen, erwartet AWS Glue keinen nennenswerten Durchsatz und führt Ihre Abfrage daher auf einem einzigen Executor aus. Um die von Ihnen bereitgestellten Daten zu begrenzen und

keine Leistungsprobleme zu verursachen, empfehlen wir Ihnen, SQL mit einer `LIMIT`-Klausel bereitzustellen.

Example Verwenden von sampleQuery ohne Partitionierung

Das folgende Codebeispiel zeigt, wie `sampleQuery` ohne Partitionierung verwendet wird.

```
//A full sql query statement.
val query = "select name from $tableName where age > 0 limit 1"
val connectionOptions = JsonOptions(Map(
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
    "password" -> password,
    "sampleQuery" -> query ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Verwendung von sampleQuery für größere Datensätze

Wenn Sie einen großen Datensatz lesen, müssen Sie möglicherweise die JDBC-Partitionierung aktivieren, um eine Tabelle parallel abzufragen. Weitere Informationen finden Sie unter the section called “Parallel aus JDBC lesen”. Um `sampleQuery` mit JDBC-Partitionierung zu verwenden, legen Sie `enablePartitioningForSampleQuery` auf wahr fest. Um dieses Feature zu aktivieren, müssen Sie einige Änderungen an Ihrer `sampleQuery` vornehmen.

Wenn Sie die JDBC-Partitionierung mit verwendensampleQuery, muss Ihre Abfrage mit „where“ oder „and“ enden, damit AWS Glue Partitionierungsbedingungen anhängen kann.

Wenn Sie die Ergebnisse Ihrer sampleQuery beim parallelen Lesen aus JDBC-Tabellen einschränken möchten, legen Sie den `"sampleSize"`-Parameter fest, anstatt eine `LIMIT`-Klausel anzugeben.

Example Verwenden sampleQuery mit JDBC-Partitionierung

Das folgende Codebeispiel zeigt, wie `sampleQuery` mit JDBC-Partitionierung verwendet wird.

```
//note that the query should end with "where" or "and" if use with JDBC partitioning.
val query = "select name from $tableName where age > 0 and"

//Enable JDBC partitioning by setting hashfield.
//to use sampleQuery with partitioning, set enablePartitioningForSampleQuery.
//use sampleSize to limit the size of returned data.
val connectionOptions = JsonOptions(Map(
```

```
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
    "password" -> password,
    "hashfield" -> primaryKey,
    "sampleQuery" -> query,
    "enablePartitioningForSampleQuery" -> true,
    "sampleSize" -> "1" ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Hinweise und Einschränkungen:

Beispielabfragen können nicht zusammen mit Auftragslesezeichen verwendet werden. Der Lesezeichenstatus wird ignoriert, wenn die Konfiguration für beide bereitgestellt wird.

Benutzerdefinierten JDBC-Treiber verwenden

In den folgenden Codebeispielen wird gezeigt, wie JDBC-Datenbanken mit benutzerdefinierten JDBC-Treibern gelesen und geschrieben werden. Diese demonstrieren das Lesen aus einer Version eines Datenbankprodukts und das Schreiben in eine spätere Version desselben Produkts.

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time
from pyspark.sql.types import StructType, StructField, IntegerType, StringType

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Construct JDBC connection options
connection_mysql5_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}
```

```
connection_mysql8_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://amzn-s3-demo-bucket/mysql-connector-
java-8.0.17.jar",
    "customJdbcDriverClassName": "com.mysql.cj.jdbc.Driver"}

connection_oracle11_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}

connection_oracle18_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://amzn-s3-demo-bucket/ojdbc10.jar",
    "customJdbcDriverClassName": "oracle.jdbc.OracleDriver"}

# Read from JDBC databases with custom driver
df_mysql8 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql8_options)

# Read DynamicFrame from MySQL 5 and write to MySQL 8
df_mysql5 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql5_options)
glueContext.write_from_options(frame_or_dfc=df_mysql5, connection_type="mysql",
                               connection_options=connection_mysql8_options)

# Read DynamicFrame from Oracle 11 and write to Oracle 18
df_oracle11 =
 glueContext.create_dynamic_frame.from_options(connection_type="oracle",

 connection_options=connection_oracle11_options)
glueContext.write_from_options(frame_or_dfc=df_oracle11, connection_type="oracle",
                               connection_options=connection_oracle18_options)
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {
  val MYSQL_5_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val MYSQL_8_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val ORACLE_11_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"
  val ORACLE_18_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"

  // Construct JDBC connection options
  lazy val mysql5JsonOption = jsonOptions(MYSQL_5_URI)
  lazy val mysql8JsonOption = customJDBCDriverJsonOptions(MYSQL_8_URI, "s3://amzn-
s3-demo-bucket/mysql-connector-java-8.0.17.jar", "com.mysql.cj.jdbc.Driver")
  lazy val oracle11JsonOption = jsonOptions(ORACLE_11_URI)
  lazy val oracle18JsonOption = customJDBCDriverJsonOptions(ORACLE_18_URI, "s3://
amzn-s3-demo-bucket/ojdbc10.jar", "oracle.jdbc.OracleDriver")

  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Read from JDBC database with custom driver
    val df_mysql8: DynamicFrame = glueContext.getSource("mysql",
 mysql8JsonOption).getDynamicFrame()

    // Read DynamicFrame from MySQL 5 and write to MySQL 8
    val df_mysql5: DynamicFrame = glueContext.getSource("mysql",
 mysql5JsonOption).getDynamicFrame()
    glueContext.getSink("mysql", mysql8JsonOption).writeDynamicFrame(df_mysql5)
```

```
    // Read DynamicFrame from Oracle 11 and write to Oracle 18
    val df_oracle11: DynamicFrame = glueContext.getSource("oracle",
 oracle11JsonOption).getDynamicFrame()
    glueContext.getSink("oracle", oracle18JsonOption).writeDynamicFrame(df_oracle11)

    Job.commit()
  }

  private def jsonOptions(url: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd"}""".stripMargin)
  }

  private def customJDBCDriverJsonOptions(url: String, customJdbcDriverS3Path:
 String, customJdbcDriverClassName: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd",
         |"customJdbcDriverS3Path": "${customJdbcDriverS3Path}",
         |"customJdbcDriverClassName" :
 "${customJdbcDriverClassName}"}""".stripMargin)
  }
}
```

Parallel aus JDBC-Tabellen lesen

Sie können die Eigenschaften Ihrer JDBC-Tabelle so einstellen, dass sie aktiviert werden AWS Glue um Daten parallel zu lesen. Wenn Sie bestimmte Eigenschaften festlegen, weisen Sie an AWS Glue um parallel SQL-Abfragen für logische Partitionen Ihrer Daten auszuführen. Sie können die Partitionierung steuern, indem Sie ein Hash-Feld oder einen Hash-Ausdruck festlegen. Sie können auch die Anzahl der parallelen Lesezugriffe steuern, die für den Zugriff auf Ihre Daten verwendet werden.

Das parallele Lesen aus JDBC-Tabellen ist eine Optimierungsmethode, die die Leistung verbessern kann. Weitere Informationen darüber, wie ermittelt werden kann, wann diese Technik geeignet ist,

finden Sie unter Reduce the amount of data scan im Leitfaden Best Practices for Performance Tuning AWS Glue for Apache Spark Jobs auf AWS Prescriptive Guidance.

Um paralleles Lesen zu ermöglichen, können Sie Schlüssel-Wert-Paare im Parameterfeld Ihrer Tabellenstruktur setzen. Verwenden Sie eine JSON-Notation, um einen Wert für das Parameterfeld Ihrer Tabelle festzulegen. Weitere Informationen zum Bearbeiten der Eigenschaften einer Tabelle finden Sie unter Tabellendetails anzeigen und verwalten. Sie können auch parallele Lesevorgänge aktivieren, wenn Sie die ETL-Methoden (Extrahieren, Transformieren und Laden) `create_dynamic_frame_from_options` und `create_dynamic_frame_from_catalog` aufrufen. Weitere Informationen über das Festlegen der Optionen dieser Methoden finden Sie unter from_options und from_catalog.

Sie können diese Methode für JDBC-Tabellen verwenden, d.h. für die meisten Tabellen, deren Grundlage ein JDBC-Datenspeicher ist. Diese Eigenschaften werden beim Lesen von Amazon-Redshift- und Amazon-S3-Tabellen ignoriert.

hashfield

Stellen Sie `hashfield` auf den Namen einer Spalte in der JDBC-Tabelle ein, die zur Aufteilung der Daten in Partitionen verwendet werden soll. Um optimale Ergebnisse zu erzielen, sollte diese Spalte eine gleichmäßige Verteilung der Werte aufweisen, um die Daten zwischen den Partitionen zu verteilen. Diese Spalte kann einen beliebigen Datentyp haben. AWS Glue generiert nicht überlappende Abfragen, die parallel ausgeführt werden, um die durch diese Spalte partitionierten Daten zu lesen. Wenn Ihre Daten beispielsweise gleichmäßig nach Monaten verteilt sind, können Sie die Spalte `month` verwenden, um die Daten für jeden Monat parallel zu lesen:

```
'hashfield': 'month'
```

AWS Glue erstellt eine Abfrage, um den Feldwert mit einer Partitionsnummer zu hashen, und führt die Abfrage für alle Partitionen parallel aus. Um Ihre eigene Abfrage zur Partitionierung einer gelesenen Tabelle zu verwenden, geben Sie `hashexpression` anstelle von `hashfield` an.

hashexpression

Setzen Sie `hashexpression` auf einen SQL-Ausdruck (entsprechend der Syntax der JDBC-Datenbankmaschine), der eine ganze Zahl zurückgibt. Ein einfacher Ausdruck ist der Name einer beliebigen numerischen Spalte in der Tabelle. AWS Glue generiert SQL-Abfragen, um die JDBC-

Daten parallel zu lesen, indem die `hashexpression` `WHERE` In-the-Klausel verwendet wird, um Daten zu partitionieren.

Verwenden Sie beispielsweise die numerische Spalte `customerID`, um Daten zu lesen, die nach Kundennummer partitioniert sind:

```
'hashexpression': 'customerID'
```

Zu haben AWS Glue um die Partitionierung zu kontrollieren, geben Sie a `hashfield` statt a `hashexpression` an.

hashpartitions

Stellen Sie `hashpartitions` auf die Anzahl der parallelen Lesezugriffe der JDBC-Tabelle ein. Wenn diese Eigenschaft nicht festgelegt ist, wird der Standardwert 7 verwendet.

Stellen Sie beispielsweise die Anzahl der parallel Lesevorgänge 5 so ein, dass AWS Glue liest Ihre Daten mit fünf Abfragen (oder weniger).

```
'hashpartitions': '5'
```

Einrichtung von Amazon VPC für JDBC-Verbindungen zu Amazon RDS-Datenspeichern von AWS Glue

Wenn Sie JDBC verwenden, um eine Verbindung zu Datenbanken in Amazon RDS herzustellen, müssen Sie zusätzliche Einstellungen vornehmen. Um zu aktivieren AWS Glue Komponenten für die Kommunikation mit Amazon RDS, Sie müssen den Zugriff auf Ihre Amazon RDS-Datenspeicher in Amazon VPC einrichten. Um zu aktivieren AWS Glue Um zwischen ihren Komponenten zu kommunizieren, geben Sie eine Sicherheitsgruppe mit einer selbstreferenzierenden Regel für eingehende Nachrichten für alle TCP-Ports an. Indem Sie eine selbstreferenzierende Regel erstellen, können Sie die Quelle auf dieselbe Sicherheitsgruppe in der VPC beschränken. Eine selbstreferenzierende Regel öffnet die VPC nicht für alle Netzwerke. Die Standardsicherheitsgruppe für Ihre VPC verfügt möglicherweise bereits über eine selbstreferenzierende eingehende Regel für den gesamten Datenverkehr (ALL Traffic).

So richten Sie den Zugriff zwischen AWS Glue- und Amazon RDS-Datenspeichern ein

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon RDS-Konsole unter https://console.aws.amazon.com/rds/.
2. Identifizieren Sie in der Amazon RDS-Konsole die Sicherheitsgruppe (n), die zur Steuerung des Zugriffs auf Ihre Amazon RDS-Datenbank verwendet werden.

   Wählen Sie im linken Navigationsbereich Datenbanken und dann aus der Liste im Hauptbereich die Instance aus, zu der Sie eine Verbindung herstellen möchten.

   Suchen Sie auf der Datenbankdetailseite auf der Registerkarte Konnektivität und Sicherheit nach VPC-Sicherheitsgruppen.
3. Identifizieren Sie anhand Ihrer Netzwerkarchitektur, welche zugehörige Sicherheitsgruppe am besten geändert werden sollte, um den Zugriff für den AWS Glue-Dienst zu ermöglichen. Speichern Sie den Namen, *`database-security-group`* damit Sie später darauf zurückgreifen können. Wenn es keine geeignete Sicherheitsgruppe gibt, folgen Sie den Anweisungen zur Bereitstellung von Zugriff auf Ihre DB-Instance in Ihrer VPC, indem Sie eine Sicherheitsgruppe erstellen, in der Amazon RDS-Dokumentation.
4. Melden Sie sich bei der an AWS Management Console und öffnen Sie die Amazon VPC-Konsole unter https://console.aws.amazon.com/vpc/.
5. Identifizieren Sie in der Amazon VPC-Konsole, wie das Update *`database-security-group`* durchgeführt werden soll.

   Wählen Sie im linken Navigationsbereich Sicherheitsgruppen aus und wählen Sie dann eine Gruppe *`database-security-group`* aus der Liste im Hauptbereich aus.
6. Identifizieren Sie die Sicherheitsgruppen-ID für*`database-security-group`*,*`database-sg-id`*. Bewahren Sie es zum future Nachschlagen auf.

   Suchen Sie auf der Seite mit den Sicherheitsgruppen-Details nach der Sicherheitsgruppen-ID.
7. Ändern Sie die Regeln für eingehenden Datenverkehr und fügen Sie eine Regel hinzu*`database-security-group`*, die sich selbst referenziert, um dies zuzulassen AWS Glue Komponenten für die Kommunikation. Fügen Sie insbesondere eine Regel hinzu oder bestätigen Sie, dass es eine Regel gibt`All TCP`, bei der Typ`TCP`, Protokoll, Portbereich alle Ports umfasst und Quelle für steht*`database-sg-id`*. Stellen Sie sicher, dass die Sicherheitsgruppe, die Sie für Source eingegeben haben, mit der Sicherheitsgruppe identisch ist, die Sie bearbeiten.

Wählen Sie auf der Detailseite der Sicherheitsgruppe die Option Regeln für eingehenden Datenverkehr bearbeiten aus.

Die eingehende Regel sollte etwa wie folgt aussehen:

Typ	Protocol (Protokoll)	Port-Bereich	Quelle
Alle TCP	TCP	0–65535	*database-sg-id*

8. Fügen Sie Regeln für ausgehenden Datenverkehr hinzu.

   Wählen Sie auf der Detailseite der Sicherheitsgruppe die Option Regeln für ausgehenden Datenverkehr bearbeiten aus.

   Wenn Ihre Sicherheitsgruppe den gesamten ausgehenden Datenverkehr zulässt, benötigen Sie keine separaten Regeln. Zum Beispiel:

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Gesamter Datenverkehr	ALL	ALL	0.0.0.0/0

   Wenn Ihre Netzwerkarchitektur so konzipiert ist, dass Sie ausgehenden Datenverkehr einschränken, erstellen Sie die folgenden Regeln für ausgehenden Datenverkehr:

   Erstellen Sie eine selbstreferenzierende Regel, wobei Typ für**`All TCP`**, Protokoll**`TCP`**, Portbereich für alle Ports und Ziel für stehen. *database-sg-id* Stellen Sie sicher, dass die Sicherheitsgruppe, die Sie für Ziel eingegeben haben, mit der Sicherheitsgruppe identisch ist, die Sie bearbeiten.

   Wenn Sie einen Amazon S3-VPC-Endpunkt verwenden, fügen Sie eine HTTPS-Regel hinzu, um Datenverkehr von der VPC zu Amazon S3 zuzulassen. Erstellen Sie eine Regel, bei der TypHTTPS, ProtokollTCP, Portbereich 443 und Destination die ID der verwalteten Präfixliste für den Amazon S3 S3-Gateway-Endpunkt ist*s3-prefix-list-id*. Weitere Informationen zu Präfixlisten und Amazon S3-Gateway-Endpunkten finden Sie unter Gateway-Endpunkte für Amazon S3 in der Amazon VPC-Dokumentation.

Zum Beispiel:

Typ	Protocol (Protokoll)	Port-Bereich	Bestimmungsort
Alle TCP	TCP	0–65535	*database-sg-id*
HTTPS	TCP	443	*s3-prefix-list-id*

MongoDB-Verbindungen

Sie können AWS Glue for Spark verwenden, um in MongoDB und MongoDB Atlas in AWS Glue 4.0 und späteren Versionen aus Tabellen zu lesen und in Tabellen zu schreiben. Sie können eine Verbindung zu MongoDB herstellen, indem Sie die Anmeldeinformationen für Benutzername und Passwort verwenden, die AWS Secrets Manager über eine AWS Glue-Verbindung gespeichert sind.

Weitere Informationen über MongoDB finden Sie in der MongoDB-Dokumentation.

Konfigurieren von MongoDB-Verbindungen

Um von AWS Glue aus eine Verbindung zu MongoDB herzustellen, benötigen Sie Ihre MongoDB-Anmeldeinformationen und*mongodbUser*. *mongodbPass*

Um von AWS Glue aus eine Verbindung zu MongoDB herzustellen, benötigen Sie möglicherweise einige Voraussetzungen:

- Wenn sich Ihre MongoDB-Instance in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue-Job mit der MongoDB-Instance kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die AWS Glue bei der Ausführung des Jobs verwendet. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer MongoDB-Instance und diesem Standort zugelassen wird. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

Sie können dann mit der Konfiguration von AWS Glue für die Verwendung mit MongoDB fortfahren.

Eine Verbindung zu MongoDB konfigurieren Sie wie folgt:

1. Erstellen Sie optional ein Geheimnis mit Ihren MongoDB-Anmeldeinformationen. AWS Secrets Manager Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `username` mit dem Wert. *mongodbUser*

     Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *mongodbPass*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für die future Verwendung in AWS Glue bei. *connectionName*

   - Wählen Sie als Verbindungstyp die Option MongoDB oder MongoDB Atlas aus.
   - Geben Sie als MongoDB-URL oder MongoDB-Atlas-URL den Hostnamen Ihrer MongoDB-Instance an.

     Eine MongoDB-URL wird im Format `mongodb://`*mongoHost*`:`*mongoPort*`/`*mongoDBname* bereitgestellt.

     Eine MongoDB-Atlas-URL wird im Format `mongodb +srv://`*mongoHost*`:`*mongoPort*`/`*mongoDBname* bereitgestellt.

     Die Bereitstellung der Standarddatenbank für die Verbindung *mongoDBname* ist optional.
   - Wenn Sie ein Secrets Manager Manager-Geheimnis erstellen möchten, wählen Sie den Typ AWS Secrets Manager Credential.

     Geben Sie dann in AWS Secret Folgendes ein*secretName*.
   - Wenn Sie sich dafür entscheiden, Benutzername und Passwort anzugeben, geben Sie *mongodbUser* und ein*mongodbPass*.

3. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für MongoDB-Instances, die AWS in einer Amazon VPC gehostet werden

- Sie müssen Amazon VPC-Verbindungsinformationen für die AWS Glue-Verbindung bereitstellen, die Ihre MongoDB-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Nachdem Sie eine AWS Glue MongoDB-Verbindung erstellt haben, müssen Sie die folgenden Aktionen ausführen, bevor Sie Ihre Verbindungsmethode aufrufen:

- Wenn Sie sich dafür entschieden haben, ein Secrets Manager Manager-Geheimnis zu erstellen, gewähren Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *`secretName`* Leserechte.
- Stellen *`connectionName`* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Um Ihre AWS Glue MongoDB-Verbindung in AWS Glue for Spark zu verwenden, geben Sie die `connectionName` Option in Ihrem Verbindungsmethodenaufruf an. Alternativ können Sie den Schritten unter folgen, the section called “Integration mit MongoDB” um die Verbindung in Verbindung mit dem AWS Glue-Datenkatalog zu verwenden.

Mit einer AWS Glue-Verbindung aus MongoDB lesen

Voraussetzungen:

- Eine MongoDB-Sammlung, aus der Sie lesen möchten. Sie benötigen Identifikationsinformationen für die Sammlung.

  Eine MongoDB-Sammlung wird durch einen Datenbanknamen und einen Sammlungsnamen identifiziert,*`mongodbName`*,*`mongodbCollection`*.
- Eine AWS Glue MongoDB-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu MongoDB zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*`connectionName`*.

Zum Beispiel:

```
mongodb_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mongodb",
```

```
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "partitioner":
 "com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner",
        "partitionerOptions.partitionSizeMB": "10",
        "partitionerOptions.partitionKey": "_id",
        "disableUpdateUri": "false",
    }
)
```

Schreiben in MongoDB-Tabellen

In diesem Beispiel werden Informationen aus einer vorhandenen DynamicFrame Datenbank in MongoDB geschrieben. *dynamicFrame*

Voraussetzungen:

- Eine MongoDB-Sammlung, in der Sie schreiben möchten. Sie benötigen Identifikationsinformationen für die Sammlung.

  Eine MongoDB-Sammlung wird durch einen Datenbanknamen und einen Sammlungsnamen identifiziert,*mongodbName*,*mongodbCollection*.
- Eine AWS Glue MongoDB-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu MongoDB zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="mongodb",
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "disableUpdateUri": "false",
        "retryWrites": "false",
```

```
    },
)
```

Lesen und Schreiben von Tabellen in MongoDB-Tabellen

In diesem Beispiel werden Informationen aus einer vorhandenen DynamicFrame Datenbank in MongoDB geschrieben. *dynamicFrame*

Voraussetzungen:

- Eine MongoDB-Sammlung, aus der Sie lesen möchten. Sie benötigen Identifikationsinformationen für die Sammlung.

  Eine MongoDB-Sammlung, in der Sie schreiben möchten. Sie benötigen Identifikationsinformationen für die Sammlung.

  Eine MongoDB-Sammlung wird durch einen Datenbanknamen und einen Sammlungsnamen identifiziert,*mongodbName*,*mongodbCollection*.
- MongoDB-Authentifizierungsinformationen und*mongodbUser*. *mongodbPassword*

Zum Beispiel:

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

```
output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
mongo_uri = "mongodb://<mongo-instanced-ip-address>:27017"
mongo_ssl_uri = "mongodb://<mongo-instanced-ip-address>:27017"
write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_mongo_options = {
    "uri": mongo_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"}

ssl_mongo_options = {
    "uri": mongo_ssl_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "ssl": "true",
    "ssl.domain_match": "false"
}

write_mongo_options = {
    "uri": write_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
}

# Get DynamicFrame from MongoDB
dynamic_frame =
 glueContext.create_dynamic_frame.from_options(connection_type="mongodb",

 connection_options=read_mongo_options)

# Write DynamicFrame to MongoDB
glueContext.write_dynamic_frame.from_options(dynamicFrame,
 connection_type="mongodb", connection_options=write_mongo_options)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DEFAULT_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val defaultJsonOption = jsonOptions(DEFAULT_URI)
  lazy val writeJsonOption = jsonOptions(WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from MongoDB
    val dynamicFrame: DynamicFrame = glueContext.getSource("mongodb",
 defaultJsonOption).getDynamicFrame()

    // Write DynamicFrame to MongoDB
    glueContext.getSink("mongodb", writeJsonOption).writeDynamicFrame(dynamicFrame)

    Job.commit()
  }

  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"mongodbName",
         |"collection":"mongodbCollection",
         |"username": "mongodbUsername",
         |"password": "mongodbPassword",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
```

```
        |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Referenz zur MongoDB-Verbindungsoption

Bezeichnet eine Verbindung mit MongoDB. Die Anschlussmöglichkeiten bei einer Quellverbindung und eine Senkenverbindung unterscheiden sich.

Diese Verbindungseigenschaften werden von Quell- und Senkenverbindungen gemeinsam genutzt:

- `connectionName` – Wird zum Lesen/Schreiben verwendet. Der Name einer AWS Glue MongoDB-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkinformationen für Ihre Verbindungsmethode bereitstellt. Wenn eine AWS Glue-Verbindung wie im vorherigen Abschnitt beschrieben konfiguriert istthe section called “Konfigurieren von MongoDB”, ersetzt das `connectionName` Bereitstellen der Verbindungsoptionen die Notwendigkeit`"uri"`, die `"username"` und `"password"` -Verbindungsoptionen bereitzustellen.
- `"uri"`: (Erforderlich) Der als `mongodb://<host>:<port>` formatierte MongoDB-Host, von dem gelesen werden soll. Wird in AWS Glue-Versionen vor AWS Glue 4.0 verwendet.
- `"connection.uri"`: (Erforderlich) Der als `mongodb://<host>:<port>` formatierte MongoDB-Host, von dem gelesen werden soll. Wird in AWS Glue 4.0 und späteren Versionen verwendet.
- `"username"`: (Erforderlich) Der MongoDB-Benutzername.
- `"password"`: (Erforderlich) Das MongoDB-Passwort.
- `"database"`: (Erforderlich) Die MongoDB-Datenbank, aus der gelesen werden soll. Diese Option kann auch in `additional_options` beim Aufruf von `glue_context.create_dynamic_frame_from_catalog` in Ihrem Auftragsskript übergeben werden.
- `"collection"`: (Erforderlich) Die MongoDB-Sammlung, aus der gelesen werden soll. Diese Option kann auch in `additional_options` beim Aufruf von `glue_context.create_dynamic_frame_from_catalog` in Ihrem Auftragsskript übergeben werden.

„connectionType“: „mongodb“ als Quelle

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "mongodb"` als Quelle:

- `"ssl"`: (Optional) Wenn `true`, wird eine SSL-Verbindung initiiert. Der Standardwert ist `false`.
- `"ssl.domain_match"`: (Optional) Wenn `true` und wenn `ssl` „`true`“ ist, wird die Prüfung der Domainübereinstimmung durchgeführt. Der Standardwert ist `true`.
- `"batchSize"`: (Optional): Die Anzahl der pro Batch zurückzugebenden Dokumente, die innerhalb des Cursors interner Batches verwendet werden.
- `"partitioner"`: (Optional): Der Klassenname des Partitionierers zum Lesen von Eingabedaten aus MongoDB. Der Konnektor stellt die folgenden Partitionierer bereit:
  - `MongoDefaultPartitioner`(Standard) (In AWS Glue 4.0 nicht unterstützt)
  - `MongoSamplePartitioner`(Erfordert MongoDB 3.2 oder höher) (Wird in AWS Glue 4.0 nicht unterstützt)
  - `MongoShardedPartitioner`(In AWS Glue 4.0 nicht unterstützt)
  - `MongoSplitVectorPartitioner`(In AWS Glue 4.0 nicht unterstützt)
  - `MongoPaginateByCountPartitioner`(In AWS Glue 4.0 nicht unterstützt)
  - `MongoPaginateBySizePartitioner`(In AWS Glue 4.0 nicht unterstützt)
  - `com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.ShardedPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.PaginateIntoPartitionsPartit`
- `"partitionerOptions"` (Optional): Optionen für den angegebenen Partitionierer. Die folgenden Optionen werden für jeden Partitionierer unterstützt:
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner`: `partitionKey`, `partitionSizeMB`
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner`: `partitionKey`, `partitionSizeMB`

  Weitere Informationen zu diesen Optionen finden Sie unter Partitioner-Konfiguration in der MongoDB-Dokumentation.

„connectionType“: „mongodb“ als Sink

Verwenden Sie die folgenden Verbindungsoptionen mit `"connectionType": "mongodb"` als Senke:

- `"ssl"`: (Optional) Wenn `true`, wird eine SSL-Verbindung initiiert. Der Standardwert ist `false`.
- `"ssl.domain_match"`: (Optional) Wenn `true` und wenn `ssl` „`true`“ ist, wird die Prüfung der Domainübereinstimmung durchgeführt. Der Standardwert ist `true`.
- `"extendedBsonTypes"`: (Optional) Wenn `true`, werden beim Schreiben von Daten in MongoDB erweiterte BSON-Typen erlaubt. Der Standardwert ist `true`.
- `"replaceDocument"`: (Optional) Wenn `true`, wird das gesamte Dokument beim Speichern von Datasets, die ein `_id`-Feld enthalten, ersetzt. Wenn `false`, werden nur Felder im Dokument aktualisiert, die mit den Feldern im Dataset übereinstimmen. Der Standardwert ist `true`.
- `"maxBatchSize"`: (Optional): Die maximale Batchgröße für Massenvorgänge bei der Datenspeicherung. Der Standardwert ist 512.
- `"retryWrites"`: (Optional): Automatische Wiederholung bestimmter Schreibvorgänge ein einziges Mal, wenn AWS Glue auf einen Netzwerkfehler stößt.

SAP-HANA-Verbindungen

Sie können AWS Glue for Spark verwenden, um in SAP HANA in AWS Glue 4.0 und späteren Versionen aus Tabellen zu lesen und in Tabellen zu schreiben. Sie können definieren, was mit einer SQL-Abfrage aus SAP-HANA gelesen werden soll. Sie stellen mithilfe von JDBC-Anmeldeinformationen, die AWS Secrets Manager über eine AWS Glue SAP HANA-Verbindung gespeichert sind, eine Verbindung zu SAP HANA her.

Weitere Informationen zu SAP HANA JDBC finden Sie in der SAP-HANA-Dokumentation.

Konfigurieren von SAP-HANA-Verbindungen

Um von AWS Glue aus eine Verbindung zu SAP HANA herzustellen, müssen Sie Ihre SAP HANA-Anmeldeinformationen erstellen und in einem AWS Secrets Manager Secret speichern und dieses Secret dann mit einer SAP HANA AWS Glue-Verbindung verknüpfen. Sie müssen die Netzwerkkonnektivität zwischen Ihrem SAP HANA-Service und AWS Glue konfigurieren.

Voraussetzungen für eine Verbindung mit SAP HANA:

- Wenn sich Ihr SAP HANA-Service in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue-Job mit dem SAP HANA-Service kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die AWS Glue bei der Ausführung des Jobs verwendet. Darüber hinaus muss

Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrem SAP-HANA-Endpunkt und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem SAP-HANA-JDBC-Port herstellen. Weitere Informationen zu SAP-HANA-Ports finden Sie in der Dokumentation von SAP HANA. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

- Es gibt keine zusätzlichen Voraussetzungen, wenn Ihr SAP-HANA-Endpunkt über das Internet zugänglich ist.

Konfigurieren Sie eine Verbindung zu SAP HANA wie folgt:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer SAP HANA-Anmeldeinformationen ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *saphanaUsername*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *saphanaPassword*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für die future Verwendung in AWS Glue bei. *connectionName*

   - Wählen Sie als Verbindungstyp SAP HANA aus.
   - Wenn Sie die SAP-HANA-URL angeben, geben Sie die URL für Ihre Instance an.

     SAP HANA JDBC URLs haben das folgende Format `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWS Glue benötigt die folgenden JDBC-URL-Parameter:

     - `databaseName` – Eine Standarddatenbank in SAP HANA, mit der eine Verbindung hergestellt werden kann.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an. *secretName*

Nachdem Sie eine AWS Glue SAP HANA-Verbindung erstellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihren AWS Glue-Job ausführen:

- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *secretName* Leserechte.
- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen aus SAP-HANA-Tabellen

Voraussetzungen:

- Eine SAP-HANA-Tabelle, aus der gelesen werden soll. Sie benötigen Identifikationsinformationen für die Tabelle.

  Eine Tabelle kann mit einem SAP-HANA-Tabellennamen und -Schemanamen im folgenden Format angegeben werden: *schemaName*.*tableName*. Der Schemaname und das Trennzeichen „.“ sind nicht erforderlich, wenn sich die Tabelle im Standardschema „public“ befindet. Nennen Sie das*tableIdentifier*. Beachten Sie, dass die Datenbank als JDBC-URL-Parameter in `connectionName` bereitgestellt wird.
- Eine AWS Glue SAP HANA-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu SAP HANA zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
saphana_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier",
    }
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Ihre zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `query` konfigurieren.

Zum Beispiel:

```
saphana_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Schreiben in SAP-HANA-Tabellen

In diesem Beispiel werden Informationen aus einer vorhandenen Datei DynamicFrame *dynamicFrame* in SAP HANA geschrieben. Wenn die Tabelle bereits Informationen enthält, gibt AWS Glue einen Fehler aus.

Voraussetzungen:

- Eine SAP-HANA-Tabelle, in die geschrieben werden soll.

  Eine Tabelle kann mit einem SAP-HANA-Tabellennamen und -Schemanamen im folgenden Format angegeben werden: *schemaName*.*tableName*. Der Schemaname und das Trennzeichen „.“ sind nicht erforderlich, wenn sich die Tabelle im Standardschema „public“ befindet. Nenn das*tableIdentifier*. Beachten Sie, dass die Datenbank als JDBC-URL-Parameter in `connectionName` bereitgestellt wird.
- SAP-HANA-Authentifizierungsinformationen. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu SAP HANA zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
options = {
    "connectionName": "connectionName",
    "dbtable": 'tableIdentifier'
}

    saphana_write = glueContext.write_dynamic_frame.from_options(
        frame=dynamicFrame,
        connection_type="saphana",
        connection_options=options
)
```

Referenz zur SAP-HANA-Verbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue SAP HANA-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkinformationen für Ihre Verbindungsmethode bereitstellt.
- `databaseName` – Wird zum Lesen/Schreiben verwendet. Gültige Werte: Namen von Datenbanken in SAP HANA. Name der Datenbank, mit der eine Verbindung hergestellt werden soll.
- `dbtable` – Zum Schreiben erforderlich, zum Lesen erforderlich, sofern `query` nicht angegeben ist. Wird für Lesen/Schreiben verwendet. Gültige Werte: Inhalt einer SQL-FROM-Klausel in SAP HANA. Identifiziert eine Tabelle in SAP HANA, zu der eine Verbindung hergestellt werden soll. Sie können auch anderes SQL als einen Tabellennamen angeben, z. B. eine Unterabfrage. Weitere Informationen finden Sie unter From clause in der Dokumentation von SAP HANA.
- `query` – Wird zum Lesen verwendet. Eine SQL-SELECT-Abfrage von SAP HANA, die definiert, was beim Lesen aus SAP HANA abgerufen werden soll.

Snowflake-Verbindungen

In AWS Glue 4.0 und späteren Versionen können Sie Glue for Spark verwenden, um aus Tabellen in Snowflake zu lesen und in AWS Tabellen zu schreiben. Sie können aus Snowflake mit einer SQL-Abfrage lesen. Sie können mit einem Benutzer und einem Passwort eine Verbindung zu Snowflake herstellen. Sie können auf Snowflake-Anmeldeinformationen verweisen, die im AWS Secrets Manager AWS Glue-Datenkatalog gespeichert sind. Data Catalog Snowflake-Anmeldeinformationen für AWS Glue for Spark werden getrennt von Data Catalog Snowflake-Anmeldeinformationen für Crawler gespeichert. Sie müssen einen `SNOWFLAKE`-Verbindungstyp auswählen und keinen `JDBC`-Verbindungstyp, der für die Verbindung mit Snowflake konfiguriert ist.

Weitere Informationen zu Snowflake finden Sie auf der Snowflake-Website. Weitere Informationen zu Snowflake finden Sie unter Snowflake Data Warehouse auf Amazon Web Services. AWS

Konfiguration von Snowflake-Verbindungen

Es gibt keine AWS Voraussetzungen für die Verbindung zu Snowflake-Datenbanken, die über das Internet verfügbar sind.

Optional können Sie die folgende Konfiguration durchführen, um Ihre Verbindungsdaten mit AWS Glue zu verwalten.

Um Ihre Verbindungsdaten mit AWS Glue zu verwalten

1. Generieren Sie in Snowflake einen Benutzer *snowflakeUser* und ein Passwort. *snowflakePassword*
2. Erstellen Sie in AWS Secrets Manager ein Geheimnis mit Ihren Snowflake-Anmeldeinformationen. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für *snowflakeUser* mit dem Schlüssel. `USERNAME`
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für *snowflakePassword* mit dem Schlüssel. `PASSWORD`
   - Bei der Auswahl von Schlüssel/Wert-Paaren können Sie Ihrem Snowflake-Warehouse den Schlüssel `sfWarehouse` bereitstellen.
3. Erstellen AWS Sie im Glue-Datenkatalog eine Verbindung, indem Sie Verbindungen und dann Verbindung erstellen wählen. Folgen Sie den Schritten im Verbindungsassistenten, um den Vorgang abzuschließen:

   - Wählen Sie bei der Auswahl einer Datenquelle Snowflake und dann Weiter aus.
   - Geben Sie die Verbindungsdetails wie Host und Port ein. Geben Sie bei der Eingabe der Snowflake-Host-Snowflake-URL die URL Ihrer Snowflake-Instanz an. Die URL verwendet normalerweise einen Hostnamen in der Form. *account_identifier*`.snowflakecomputing.com` Das URL-Format kann jedoch je nach Ihrem Snowflake-Kontotyp variieren (z. B. Azure oder von Snowflake gehostet). AWS
   - Wählen Sie bei der Auswahl der IAM-Dienstrolle aus dem Drop-down-Menü aus. Dies ist die IAM-Rolle aus Ihrem Konto, die für den Zugriff auf AWS Secrets Manager und die Zuweisung von IP-Adressen verwendet wird, wenn VPC angegeben ist.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an. *secretName*
4. Legen Sie im nächsten Schritt des Assistenten die Eigenschaften für Ihre Snowflake-Verbindung fest.
5. Überprüfen Sie im letzten Schritt des Assistenten Ihre Einstellungen und schließen Sie dann den Vorgang ab, um Ihre Verbindung herzustellen.

In den folgenden Situationen benötigen Sie möglicherweise Folgendes:

- Für Snowflake, gehostet auf einer AWS Amazon VPC
  - Sie benötigen eine entsprechende Amazon-VPC-Konfiguration für Snowflake. Weitere Informationen zur Konfiguration Ihrer Amazon VPC finden Sie unter AWS PrivateLink & Snowflake in der Snowflake-Dokumentation.
  - Sie benötigen eine entsprechende Amazon VPC-Konfiguration für AWS Glue. the section called “Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS PrivateLink für AWS Glue ”.
  - Sie müssen eine AWS Glue Data Catalog-Verbindung erstellen, die Amazon VPC-Verbindungsinformationen bereitstellt (zusätzlich zur ID eines AWS Secrets Manager Secrets, das Ihre Snowflake-Sicherheitsanmeldedaten definiert). Ihre URL ändert sich bei der Verwendung AWS PrivateLink, wie in der Snowflake-Dokumentation beschrieben, die in einem vorherigen Artikel verlinkt wurde.
  - Sie müssen in Ihrer Auftragskonfiguration die Verbindung zu Data Catalog als zusätzliche Netzwerkverbindung einschließen.

Lesen aus Snowflake-Tabellen

Voraussetzungen: Eine Snowflake-Tabelle, aus der Sie gerne lesen möchten. Sie benötigen den Namen der Snowflake-Tabelle,. *tableName* Sie benötigen Ihre Snowflake-URL*snowflakeUrl*, Ihren Benutzernamen und Ihr Passwort. *snowflakeUser* *snowflakePassword* Wenn Ihr Snowflake-Benutzer keinen Standard-Namespace festgelegt hat, benötigen Sie den Snowflake-Datenbanknamen und den Schemanamen. *databaseName* *schemaName* Wenn Ihr Snowflake-Benutzer kein Standard-Warehouse-Set hat, benötigen Sie außerdem einen Warehouse-Namen. *warehouseName*

Zum Beispiel:

Zusätzliche Voraussetzungen: Führen Sie die Schritte zur Verwaltung Ihrer Verbindungsdaten mit AWS Glue aus, um sie zu konfigurieren*snowflakeUrl*, *snowflakeUsername* und*snowflakePassword*. Weitere Informationen zu diesen Schritten finden Sie im vorherigen Abschnitt unter the section called “Snowflake konfigurieren”. Um auszuwählen, mit welcher zusätzlichen Netzwerkverbindung eine Verbindung hergestellt werden soll, verwenden wir den `connectionName`-Parameter.

```
snowflake_read = glueContext.create_dynamic_frame.from_options(
  connection_type="snowflake",
  connection_options={
```

```
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

Darüber hinaus können Sie die Parameter `autopushdown` und `query` verwenden, um einen Teil einer Snowflake-Tabelle zu lesen. Dies kann wesentlich effizienter sein als das Filtern Ihrer Ergebnisse, nachdem diese in Spark geladen wurden. Stellen Sie sich ein Beispiel vor, bei dem alle Verkäufe in derselben Tabelle gespeichert sind, Sie jedoch nur die Verkäufe eines bestimmten Geschäfts an Feiertagen analysieren müssen. Wenn diese Informationen in der Tabelle gespeichert sind, können Sie mithilfe des Prädikat-Pushdowns die Ergebnisse wie folgt abrufen:

```
snowflake_node = glueContext.create_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "autopushdown": "on",
        "query": "select * from sales where store='1' and IsHoliday='TRUE'",
        "connectionName": "snowflake-glue-conn",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

Schreiben in Snowflake-Tabellen

Voraussetzungen: Eine Snowflake-Datenbank, in die Sie schreiben möchten. Sie benötigen einen aktuellen oder gewünschten Tabellennamen,*tableName*. Sie benötigen Ihre Snowflake-URL*snowflakeUrl*, Ihren Benutzernamen *snowflakeUser* und Ihr Passwort. *snowflakePassword* Wenn Ihr Snowflake-Benutzer keinen Standard-Namespace festgelegt hat, benötigen Sie den Snowflake-Datenbanknamen und den Schemanamen. *databaseName* *schemaName* Wenn Ihr Snowflake-Benutzer kein Standard-Warehouse-Set hat, benötigen Sie außerdem einen Warehouse-Namen. *warehouseName*

Zum Beispiel:

Zusätzliche Voraussetzungen: Führen Sie die Schritte zur Verwaltung Ihrer Verbindungsdaten mit AWS Glue aus, um sie zu konfigurieren*snowflakeUrl*, *snowflakeUsername*

und*snowflakePassword*. Weitere Informationen zu diesen Schritten finden Sie im vorherigen Abschnitt unter the section called “Snowflake konfigurieren”. Um auszuwählen, mit welcher zusätzlichen Netzwerkverbindung eine Verbindung hergestellt werden soll, verwenden wir den `connectionName`-Parameter.

```
glueContext.write_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    },
)
```

Referenz zur Snowflake-Verbindungsoption

Der Snowflake-Verbindungstyp akzeptiert die folgenden Verbindungsoptionen:

Sie können einige der Parameter in diesem Abschnitt über eine Data-Catalog-Verbindung (`sfUrl`, `sfUser`, `sfPassword`) abrufen. In diesem Fall müssen Sie sie nicht angeben. Sie können dies tun, indem Sie den Parameter `connectionName` angeben.

Sie können einige der Parameter in diesem Abschnitt aus einem AWS Secrets Manager geheimen Schlüssel (`sfUser`,`sfPassword`) abrufen. In diesem Fall müssen Sie sie nicht angeben. Das Secret muss den Inhalt unter den `sfUser`- und `sfPassword`-Schlüsseln bereitstellen. Sie können dies tun, indem Sie den Parameter `secretId` angeben.

Die folgenden Parameter werden im Allgemeinen beim Herstellen einer Verbindung mit Snowflake verwendet.

- `sfDatabase` – Erforderlich, wenn in Snowflake kein Benutzerstandard festgelegt ist. Wird für Lesen/Schreiben verwendet. Die Datenbank, die nach dem Herstellen der Verbindung für die Sitzung verwendet werden soll.
- `sfSchema` – Erforderlich, wenn in Snowflake kein Benutzerstandard festgelegt ist. Wird für Lesen/Schreiben verwendet. Das Schema, das für die Sitzung nach dem Herstellen der Verbindung verwendet werden soll.

- `sfWarehouse` – Erforderlich, wenn in Snowflake kein Benutzerstandard festgelegt ist. Wird für Lesen/Schreiben verwendet. Das standardmäßige virtuelle Warehouse, das für die Sitzung nach dem Herstellen der Verbindung verwendet wird.
- `sfRole` – Erforderlich, wenn in Snowflake kein Benutzerstandard festgelegt ist. Wird für Lesen/Schreiben verwendet. Die Standardsicherheitsrolle, die nach dem Herstellen der Verbindung für die Sitzung verwendet wird.
- `sfUrl` – (Erforderlich) Wird zum Lesen/Schreiben verwendet. Gibt den Hostnamen für Ihr Konto im folgenden Format an: *`account_identifier`*`.snowflakecomputing.com`. Weitere Informationen zu Konto-IDs finden Sie unter Kontobezeichner in der Snowflake-Dokumentation.
- `sfUser` – (Erforderlich) Wird zum Lesen/Schreiben verwendet. Anmeldename für den Snowflake-Benutzer.
- `sfPassword` – (Erforderlich, außer `pem_private_key` wurde angegeben). Wird zum Lesen/Schreiben verwendet. Passwort für den Snowflake-Benutzer.
- `dbtable` – Erforderlich beim Arbeiten mit vollständigen Tabellen. Wird für Lesen/Schreiben verwendet. Der Name der zu lesenden Tabelle bzw. der Tabelle, in die Daten geschrieben werden. Beim Lesen werden alle Spalten und Datensätze abgerufen.
- `pem_private_key` – Wird zum Lesen/Schreiben verwendet. Eine unverschlüsselte, b64-kodierte Private-Key-Zeichenfolge. Der private Schlüssel für den Snowflake-Benutzer. Es ist üblich, dies aus einer PEM-Datei zu kopieren. Weitere Informationen finden Sie unter Schlüsselpaar-Authentifizierung und Schlüsselpaar-Rotation in der Snowflake-Dokumentation.
- `query` – Beim Lesen mit einer Abfrage erforderlich. Wird zum Lesen verwendet. Die genaue Abfrage (`SELECT`-Anweisung), die ausgeführt werden soll

Die folgenden Optionen werden zum Konfigurieren spezifischer Verhaltensweisen während des Verbindungsvorgangs mit Snowflake verwendet.

- `preactions` – Wird zum Lesen/Schreiben verwendet. Gültige Werte: Durch Semikolons getrennte Liste von SQL-Anweisungen als Zeichenfolge. SQL-Anweisungen werden ausgeführt, bevor Daten zwischen AWS Glue und Snowflake übertragen werden. Wenn eine Anweisung `%s` enthält, wird `%s` durch den Tabellennamen ersetzt, auf den für die Operation verwiesen wird.
- `postactions` – Wird zum Lesen/Schreiben verwendet. SQL-Anweisungen werden ausgeführt, nachdem Daten zwischen AWS Glue und Snowflake übertragen wurden. Wenn eine Anweisung `%s` enthält, wird `%s` durch den Tabellennamen ersetzt, auf den für die Operation verwiesen wird.
- `autopushdown` – Standardwert: `"on"`. Zulässige Werte: `"on"`, `"off"`. Dieser Parameter steuert, ob das automatische Abfrage-Pushdown aktiviert ist. Wenn Pushdown aktiviert ist, wird

bei der Ausführung einer Abfrage auf Spark ein Teil der Abfrage auf den Snowflake-Server „heruntergeschoben“, wenn dies möglich ist. Dies verbessert die Leistung einiger Abfragen. Informationen darüber, ob Ihre Abfrage per Pushdown verschoben werden kann, finden Sie unter Pushdown in der Snowflake-Dokumentation.

Darüber hinaus werden einige der auf dem Snowflake Spark-Konnektor verfügbaren Optionen möglicherweise in AWS Glue unterstützt. Weitere Informationen zu den Optionen, die im Snowflake-Spark-Konnektor verfügbar sind, finden Sie unter Festlegen von Konfigurationsoptionen für den Konnektor in der Snowflake-Dokumentation.

Einschränkungen des Snowflake-Konnektors

Die Verbindung zu Snowflake mit AWS Glue for Spark unterliegt den folgenden Einschränkungen.

- Dieser Konnektor unterstützt keine Auftragslesezeichen. Weitere Informationen zu Auftragslesezeichen finden Sie unter the section called “Verfolgen von verarbeiteten Daten mit Auftragslesezeichen”.
- Dieser Konnektor unterstützt keine Snowflake-Lese- und Schreibvorgänge durch Tabellen im AWS Glue-Datenkatalog mit den Methoden `create_dynamic_frame.from_catalog` und`write_dynamic_frame.from_catalog`.
- Dieser Konnektor unterstützt keine Verbindung zu Snowflake mit anderen Anmeldeinformationen als Benutzer und Passwort.
- Dieser Konnektor wird in Streaming-Aufträgen nicht unterstützt.
- Dieser Konnektor unterstützt anweisungsbasierte `SELECT`-Abfragen beim Abrufen von Informationen (wie z. B. mit dem `query`-Parameter). Andere Arten von Abfragen (z. B. `SHOW`, `DESC` oder DML-Anweisungen) werden nicht unterstützt.
- Snowflake begrenzt die Größe des über Snowflake-Clients übermittelten Abfragetextes (d. h. SQL-Anweisungen) auf 1 MB pro Anweisung. Weitere Einzelheiten finden Sie unter Beschränkungen der Abfragetextgröße.

Teradata-Vantage-Verbindungen

Sie können AWS Glue for Spark verwenden, um in Teradata Vantage in AWS Glue 4.0 und späteren Versionen aus vorhandenen Tabellen zu lesen und in diese zu schreiben. Sie können definieren, was mit einer SQL-Abfrage aus Teradata gelesen werden soll. Sie können eine Verbindung zu Teradata herstellen, indem Sie Benutzernamen und Passwörter verwenden, die AWS Secrets Manager über eine AWS Glue-Verbindung gespeichert sind.

Weitere Informationen zu Teradata finden Sie in der Teradata-Dokumentation

Konfigurieren von Teradata-Verbindungen

Um von AWS Glue aus eine Verbindung zu Teradata herzustellen, müssen Sie Ihre Teradata-Anmeldeinformationen erstellen und in einem AWS Secrets Manager Secret speichern und dieses Secret dann mit einer AWS Glue Teradata-Verbindung verknüpfen. Wenn sich Ihre Teradata-Instance in einer Amazon VPC befindet, müssen Sie auch Netzwerkoptionen für Ihre AWS Glue Teradata-Verbindung bereitstellen.

Um von AWS Glue aus eine Verbindung zu Teradata herzustellen, benötigen Sie möglicherweise einige Voraussetzungen:

- Wenn Sie über Amazon VPC auf Ihre Teradata-Umgebung zugreifen, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue-Job mit der Teradata-Umgebung kommunizieren kann. Wir raten davon ab, über das öffentliche Internet auf die Teradata-Umgebung zuzugreifen.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die AWS Glue bei der Ausführung des Jobs verwendet. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer Teradata-Instance und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem Teradata-Client-Port herstellen. Weitere Informationen zu Teradata-Ports finden Sie in der Teradata-Dokumentation.

  Abhängig von Ihrem Netzwerklayout kann eine sichere VPC-Konnektivität Änderungen an Amazon VPC und anderen Netzwerkdiensten erfordern. Weitere Informationen zur Konnektivität finden Sie unter AWS Konnektivitätsoptionen in der AWS Teradata-Dokumentation.

So konfigurieren Sie eine AWS Glue Teradata-Verbindung:

1. Identifizieren oder erstellen Sie in Ihrer Teradata-Konfiguration einen Benutzer und ein Passwort, mit dem AWS Glue eine Verbindung herstellt, *`teradataUser`* und. *`teradataPassword`* Weitere Informationen finden Sie in der Teradata-Dokumentation unter Vantage Security Overview.
2. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer Teradata-Anmeldeinformationen ein Geheimnis. Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *`secretName`* für den nächsten Schritt bei.

- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *`teradataUsername`*
- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *`teradataPassword`*

3. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *`connectionName`*

   - Wählen Sie als Verbindungstyp Teradata aus.
   - Wenn Sie die IDBC-URL angeben, geben Sie die URL für Ihre Instance an. Sie können auch bestimmte durch Kommas getrennte Verbindungsparameter in Ihrer JDBC-URL fest codieren. Die URL muss dem folgenden Format entsprechen: `jdbc:teradata://`*`teradataHostname`*`/`*`ParameterName`*`=`*`ParameterValue`*`,`*`ParameterName`*

     Zu den unterstützten URL-Parametern gehören:

     - `DATABASE` – Name der Datenbank auf dem Host, auf die standardmäßig zugegriffen werden soll.
     - `DBS_PORT` – der Datenbankport, der verwendet wird, wenn nicht der standardmäßige Port genutzt wird.
   - Wählen Sie bei der Auswahl eines Anmeldeinformationstyps die Option AWS Secrets Managerund setzen Sie dann AWS Geheim auf*`secretName`*.

4. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für Teradata-Instances, die AWS in einer Amazon VPC gehostet werden
     - Sie müssen Amazon VPC-Verbindungsinformationen für die AWS Glue-Verbindung bereitstellen, die Ihre Teradata-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Nachdem Sie eine AWS Glue Teradata-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihre Verbindungsmethode aufrufen.

- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *`secretName`* Leserechte.

- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen aus Teradata

Voraussetzungen:

- Eine Teradata-Tabelle, aus der gelesen werden soll. Sie benötigen den Tabellennamen,*tableName*.
- Eine AWS Glue Teradata-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie folgt vor, um eine Verbindung zu Teradata zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.

Zum Beispiel:

```
teradata_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Ihre zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen `query` konfigurieren.

Zum Beispiel:

```
teradata_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Schreiben in Teradata-Tabellen

Voraussetzungen: Eine Teradata-Tabelle, in die Sie schreiben möchten,. *tableName* Sie müssen die Tabelle erstellen, bevor Sie die Verbindungsmethode aufrufen.

Zum Beispiel:

```
teradata_write = glueContext.write_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
)
```

Referenz zur Teradata-Verbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue Teradata-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkinformationen für Ihre Verbindungsmethode bereitstellt.
- `dbtable` – Zum Schreiben erforderlich, zum Lesen erforderlich, sofern `query` nicht angegeben ist. Wird für Lesen/Schreiben verwendet. Der Name einer Tabelle, mit der Ihre Verbindungsmethode interagieren wird.
- `query` – Wird zum Lesen verwendet. Eine SELECT-SQL-Abfrage, die definiert, was beim Lesen aus Teradata abgerufen werden soll.

Teradata Vantage NOS-Verbindungen

Die Teradata NOS-Verbindung (Native Object Store) ist eine neue Verbindung für Teradata Vantage, die die Teradata WRITE_NOS-Abfrage zum Lesen aus vorhandenen Tabellen und die READ_NOS-Abfrage zum Schreiben in Tabellen nutzt. Diese Abfragen verwenden Amazon S3 als Staging-Verzeichnis. Daher ist der Teradata NOS-Connector schneller als der bestehende Teradata-Connector (JDBC-basiert), insbesondere bei der Verarbeitung großer Datenmengen.

Sie können die Teradata NOS-Verbindung in Spark verwenden, um aus bestehenden Tabellen in AWS Glue Teradata Vantage 5.0 und späteren Versionen zu lesen und in diese zu schreiben. AWS Glue Sie können definieren, was mit einer SQL-Abfrage aus Teradata gelesen werden soll. Sie können eine Verbindung zu Teradata herstellen, indem Sie Benutzernamen und Passwörter verwenden, die in einer Verbindung gespeichert sind. AWS Secrets Manager AWS Glue

Weitere Informationen zu Teradata finden Sie in der Teradata-Dokumentation.

Themen

- Eine Teradata NOS-Verbindung erstellen
- Aus Teradata-Tabellen lesen
- Schreiben in Teradata-Tabellen
- Referenz zur Teradata-Verbindungsoption
- Stellen Sie Optionen in der AWS Glue Visual ETL-Benutzeroberfläche bereit

Eine Teradata NOS-Verbindung erstellen

Um von aus eine Verbindung zu Teradata NOS herzustellen AWS Glue, müssen Sie Ihre Teradata-Anmeldeinformationen in einem AWS Secrets Manager geheimen Ordner erstellen und speichern und diesen geheimen Schlüssel dann einer Teradata NOS-Verbindung zuordnen. AWS Glue Wenn sich Ihre Teradata-Instance in einer Amazon VPC befindet, müssen Sie auch Netzwerkoptionen für Ihre AWS Glue Teradata NOS-Verbindung bereitstellen.

Voraussetzungen:

- Wenn Sie über Amazon VPC auf Ihre Teradata-Umgebung zugreifen, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue Job mit der Teradata-Umgebung kommunizieren kann. Wir raten davon ab, über das öffentliche Internet auf die Teradata-Umgebung zuzugreifen.
- Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die bei der Ausführung des Jobs verwendet AWS Glue werden. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer Teradata-Instance und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem Teradata-Client-Port herstellen. Weitere Informationen zu Teradata-Ports finden Sie in den Sicherheitsgruppen für Teradata Vantage.
- Abhängig von Ihrem Netzwerklayout kann eine sichere VPC-Konnektivität Änderungen an Amazon VPC und anderen Netzwerkdiensten erfordern. Weitere Informationen zur Konnektivität finden Sie unter AWS Konnektivitätsoptionen in der AWS Teradata-Dokumentation.

So konfigurieren Sie eine AWS Glue Teradata NOS-Verbindung:

1. Identifizieren oder erstellen Sie in Ihrer Teradata-Konfiguration eine *teradataUsername* und *teradataPassword* AWS Glue stellen Sie eine Verbindung her mit. Weitere Informationen finden Sie unter Vantage Security Overview in der Teradata-Dokumentation.
2. Erstellen Sie unter mit Ihren Teradata-Anmeldeinformationen einen geheimen Schlüssel. AWS Secrets Manager Um ein Geheimnis zu erstellen AWS Secrets Manager, folgen Sie dem Tutorial, das in der Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist AWS Secrets Manager. Nachdem Sie das Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel USERNAME mit dem Wert. *teradataUsername*
   - Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel PASSWORD mit dem Wert. *teradataPassword*
3. Stellen Sie in der AWS Glue Konsole eine Verbindung her, indem Sie den Schritten unter Verbindung hinzufügen folgen. AWS Glue Behalten Sie nach dem Erstellen der Verbindung den Verbindungsnamen für den nächsten Schritt bei. *connectionName*

   - Wählen Sie bei der Auswahl eines Verbindungstyps Teradata Vantage NOS aus.
   - Wenn Sie die IDBC-URL angeben, geben Sie die URL für Ihre Instance an. Sie können auch bestimmte durch Kommas getrennte Verbindungsparameter in Ihrer JDBC-URL fest codieren. Die URL muss dem folgenden Format entsprechen:. `jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName=ParameterValue`
   - Zu den unterstützten URL-Parametern gehören:
     - `DATABASE` – Name der Datenbank auf dem Host, auf die standardmäßig zugegriffen werden soll.
     - `DBS_PORT` – der Datenbankport, der verwendet wird, wenn nicht der standardmäßige Port genutzt wird.
   - Wählen Sie bei der Auswahl eines Anmeldeinformationstyps die Option AWS Secrets Manager und setzen Sie dann AWS Geheim auf. *secretName*
4. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für Teradata-Instances, die auf AWS einer Amazon VPC gehostet werden, müssen Sie Amazon VPC-Verbindungsinformationen für die Verbindung bereitstellen, die Ihre AWS

Glue Teradata-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC -, Subnetz - und Sicherheitsgruppen in den Netzwerkoptionen fest.

Nachdem Sie eine AWS Glue Teradata Vantage NOS-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihre Verbindungsmethode aufrufen.

1. Erteilen Sie der mit Ihrem AWS Glue Job verknüpften IAM-Rolle Leserechte. *secretName*
2. Stellen *connectionName* Sie in Ihrer AWS Glue Jobkonfiguration unter Verbindungen eine zusätzliche Netzwerkverbindung bereit.

Aus Teradata-Tabellen lesen

Voraussetzungen:

- Eine Teradata-Tabelle, aus der gelesen werden soll. Sie benötigen den Tabellennamen,. *tableName*
- Die Teradata-Umgebung hat Schreibzugriff auf den Amazon S3 S3-Pfad, der in der `staging_fs_url` Option, angegeben ist. *stagingFsUrl*
- Die mit dem AWS Glue Job verknüpfte IAM-Rolle hat Schreibzugriff auf den Amazon S3 S3-Standort, der in der `staging_fs_url` Option angegeben ist.
- Eine AWS Glue Teradata NOS-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie folgt vorSo konfigurieren Sie eine AWS Glue Teradata NOS-Verbindung:, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue Verbindung,*connectionName*.

Beispiel:

```
teradata_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type= "teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Ihre zurückgegebenen Ergebnisse zu filtern DynamicFrame. Sie müssen die Abfrage konfigurieren. Wenn Sie sowohl DbTable als auch Query konfigurieren, kann der Connector die Daten nicht lesen. Zum Beispiel:

```
teradata_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "query": "query",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

Darüber hinaus können Sie die DataFrame Spark-API verwenden, um aus Teradata-Tabellen zu lesen. Zum Beispiel:

```
options = {
    "url": "JDBC_URL",
    "dbtable": "tableName",
    "user": "teradataUsername", # or use "username" as key here
    "password": "teradataPassword",
    "staging_fs_url": "stagingFsUrl"
}
teradata_read_table = spark.read.format("teradatanos").option(**options).load()
```

Schreiben in Teradata-Tabellen

Voraussetzungen

- Eine Teradata-Tabelle, in die Sie schreiben möchten:. *tableName*
- Die Teradata-Umgebung hat Lesezugriff auf den Amazon S3 S3-Standort, der in der `staging_fs_url` Option, angegeben ist. *stagingFsUrl*
- Die mit dem AWS Glue Job verknüpfte IAM-Rolle hat Schreibzugriff auf den Amazon S3 S3-Standort, der in der `staging_fs_url` Option angegeben ist.
- Eine AWS Glue Teradata-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie unter beschrieben vorSo konfigurieren Sie eine AWS Glue Teradata NOS-Verbindung:, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue Verbindung,*connectionName*.

Zum Beispiel:

```
teradata_write = glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type= "teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

Referenz zur Teradata-Verbindungsoption

Verbindungs- und Betriebsoptionen:

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue Teradata-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkinformationen für Ihre Verbindungsmethode bereitstellt.
- `staging_fs_url` – Erforderlich. Wird für Lesen/Schreiben verwendet. Ein beschreibbarer Speicherort in Amazon S3, der für entladene Daten beim Lesen aus Teradata und für Parquet-Daten, die beim Schreiben in Teradata in Redshift geladen werden sollen, verwendet wird. Der S3-Bucket muss sich in derselben Region wie die Region Ihrer Jobs befinden. AWS Glue
- `dbtable` – Zum Schreiben erforderlich, zum Lesen erforderlich, sofern `query` nicht angegeben ist. Wird für Lesen/Schreiben verwendet. Der Name einer Tabelle, mit der Ihre Verbindungsmethode interagieren wird.
- `query` – Wird zum Lesen verwendet. Eine SELECT-SQL-Abfrage, die definiert, was beim Lesen aus Teradata abgerufen werden soll. Sie können nicht bestehen, wenn `dbtable` die Option angegeben ist.
- `clean_staging_s3_dir`— Fakultativ. Wird für Lesen/Schreiben verwendet. Falls wahr, bereinigen Sie die Amazon S3 S3-Staging-Objekte nach einem Lese- oder Schreibvorgang. Der Standardwert ist "True".
- `pre_actions`— Fakultativ. Wird zum Schreiben verwendet. Durch Semikolons getrennte Liste von SQL-Befehlen, die ausgeführt werden, bevor Daten zwischen Spark und Teradata Vantage übertragen werden.

- `post_actions`— Fakultativ. Wird zum Schreiben verwendet. Durch Semikolons getrennte Liste von SQL-Befehlen, die ausgeführt werden, nachdem Daten zwischen Spark und Teradata Vantage übertragen wurden.
- `truncate`— Fakultativ. Wird zum Schreiben verwendet. Falls wahr, kürzt der Konnektor die Tabelle, wenn im Überschreibmodus geschrieben wird. Wenn der Wert falsch ist, löscht der Konnektor die Tabelle, wenn im Überschreibmodus geschrieben wird. Der Standardwert ist "false".
- `create_table_script`— Fakultativ. Wird zum Schreiben verwendet. Eine SQL-Anweisung zum Erstellen einer Tabelle beim Schreiben in Teradata Vantage. Nützlich, wenn Sie eine Tabelle mit benutzerdefinierten Metadaten erstellen möchten (z. B. CREATE MULTISET oder SET table oder Änderung des Primärindex). Beachten Sie, dass der im Skript zum Erstellen einer Tabelle verwendete Tabellenname mit dem in der `dbtable` Option angegebenen Tabellennamen übereinstimmen sollte.
- `partition_size_in_mb`— Fakultativ. Wird zum Lesen verwendet. Maximale Größe einer Spark-Partition in Megabyte beim Lesen von Amazon S3 S3-Staging-Objekten. Der Standardwert lautet 128.

Sie können bei der Erstellung eines Teradata-Knotens erweiterte Optionen angeben. Diese Optionen sind dieselben wie bei der Programmierung AWS Glue für Spark-Skripte.

Siehe the section called “Teradata-Vantage-Verbindungen”.

Autorisierungsoptionen:

Im Folgenden finden Sie Optionen zur Bereitstellung von AWS Kontoanmeldeinformationen, die der Connector für den Zugriff auf den Amazon S3 S3-Staging-Bucket verwendet. Sie können wählen, (1) überhaupt keine Autorisierungsoptionen bereitzustellen und temporäre Anmeldeinformationen zu verwenden, die von Ihrer AWS Glue Ausführungsrolle generiert wurden; oder (2) ein von Ihnen erstelltes Autorisierungsobjekt bereitzustellen; oder (3) anzugeben, `auth_object aws_access_key_id and aws_secret_access_key` falls langfristige Anmeldeinformationen verwendet werden, oder, `aws_access_keyaws_secret_access_key`, bereitzustellen und `aws_session_token` temporäre Anmeldeinformationen zu verwenden.

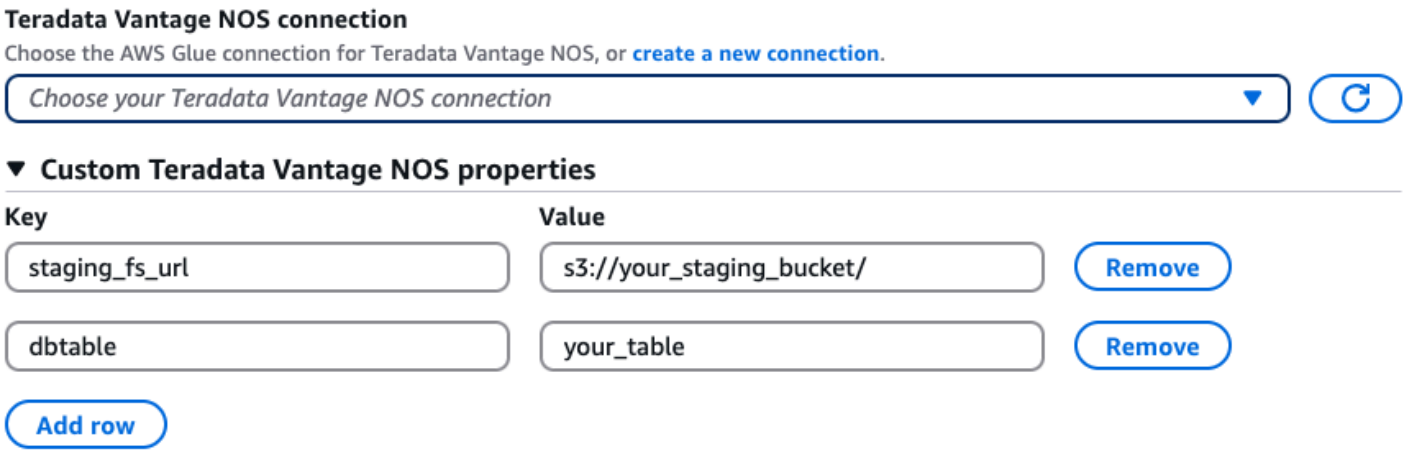
- `auth_object` Optional. Wird für den Zugriff auf den Amazon S3 S3-Staging-Bucket verwendet. Eine Zeichenfolge für ein Autorisierungsobjekt, die in einer Teradata-Instance erstellt wurde. Falls angegeben, verwendet der Connector dieses Autorisierungsobjekt, um auf den Amazon S3 S3-Staging-Bucket zuzugreifen. Falls nicht angegeben `aws_access_key_id` und auch nicht angegeben, `aws_secret_access_key` werden temporäre Anmeldeinformationen aus

der AWS Glue Ausführungsrolle abgerufen und vom Connector verwendet. Das mit diesem Autorisierungsobjekt verknüpfte AWS Konto muss sich in derselben Region wie Ihre AWS Glue Jobs und Ihr Amazon S3 S3-Staging-Bucket befinden oder mit kontoübergreifender Vertrauensstellung konfiguriert sein.

- `aws_access_key_id` Optional. Wird für den Zugriff auf den Amazon S3 S3-Staging-Bucket verwendet. Teil eines Sicherheitsnachweises für ein AWS Konto. Wenn `auth_object` nicht angegeben und mit bereitgestellt `aws_access_key_id` wird, verwendet der Connector sie`aws_secret_access_key`, um auf den Amazon S3 S3-Staging-Bucket zuzugreifen. Das mit diesem Zugriffsschlüssel verknüpfte AWS Konto muss sich in derselben Region wie Ihre AWS Glue Jobs und Ihr Amazon S3 S3-Staging-Bucket befinden oder mit kontoübergreifender Vertrauensstellung konfiguriert sein.
- `aws_secret_access_key` Optional. Wird für den Zugriff auf den Amazon S3 S3-Staging-Bucket verwendet. Teil eines Sicherheitsnachweises für ein AWS Konto. Wenn `auth_object` nicht angegeben und mit bereitgestellt `aws_secret_access_key` wird, verwendet der Connector sie`aws_access_key_id`, um auf den Amazon S3 S3-Staging-Bucket zuzugreifen. Das mit diesem geheimen Schlüssel verknüpfte AWS Konto muss sich in derselben Region wie Ihre AWS Glue Jobs und Ihr Amazon S3 S3-Staging-Bucket befinden oder mit kontoübergreifender Vertrauensstellung konfiguriert sein.
- `aws_session_token` Optional. Wird für den Zugriff auf den Amazon S3 S3-Staging-Bucket verwendet. Teil eines temporären AWS Konto-Sicherheitsnachweises. Sollte mit `aws_access_key_id` und `aws_secret_access_key` versehen werden.

Stellen Sie Optionen in der AWS Glue Visual ETL-Benutzeroberfläche bereit

Sie können alle oben genannten Optionen in Ihrer visuellen ETL-Job-Benutzeroberfläche bereitstellen. Für die Option ConnectionName sollten Sie sie aus der Dropdownliste der Teradata Vantage NOS-Verbindung auswählen. Für alle anderen Optionen sollten Sie sie über die benutzerdefinierten Teradata Vantage NOS-Eigenschaften als Schlüsselwertpaare angeben.



Vertica-Verbindungen

In AWS Glue 4.0 und späteren Versionen können Sie Glue for Spark verwenden, um aus Tabellen in Vertica zu lesen und in AWS Tabellen zu schreiben. Sie können definieren, was mit einer SQL-Abfrage aus Vertica gelesen werden soll. Sie stellen eine Verbindung zu Vertica her, indem Sie Benutzername und Passwort verwenden, die AWS Secrets Manager über eine AWS Glue-Verbindung gespeichert sind.

Weitere Informationen zu Vertica finden Sie in der Vertica-Dokumentation.

Konfigurieren von Vertica-Verbindungen

Um von AWS Glue aus eine Verbindung zu Vertica herzustellen, müssen Sie Ihre Vertica-Anmeldeinformationen erstellen und in einem AWS Secrets Manager geheimen Ordner speichern und dieses Geheimnis dann mit einer Vertica AWS Glue-Verbindung verknüpfen. Wenn sich Ihre Vertica-Instance in einer Amazon VPC befindet, müssen Sie auch Netzwerkoptionen für Ihre AWS Glue Vertica-Verbindung bereitstellen. Sie benötigen einen Amazon-S3-Bucket oder -Ordner zur temporären Speicherung beim Lesen und Schreiben in die Datenbank.

Um von AWS Glue aus eine Verbindung zu Vertica herzustellen, benötigen Sie einige Voraussetzungen:

- Ein Amazon S3 S3-Bucket oder -Ordner, der als temporärer Speicher verwendet wird, wenn aus der Datenbank gelesen und in die Datenbank geschrieben wird, auf den von `tempS3Path`.

> **Note**
>
> Wenn Sie Vertica in der Vorschau von AWS Glue-Auftragsdaten verwenden, werden temporäre Dateien möglicherweise nicht automatisch entfernt. `tempS3Path` Um sicherzustellen, dass temporäre Dateien entfernt werden, beenden Sie die Datenvorschau-Sitzung direkt, indem Sie im Bereich Datenvorschau die Option Sitzung beenden wählen. Wenn Sie nicht garantieren können, dass die Datenvorschau-Sitzung direkt beendet wird, sollten Sie die Amazon-S3-Lifecycle-Konfiguration so einrichten, dass alte Daten entfernt werden. Wir empfehlen, Daten zu entfernen, die älter als 49 Stunden sind, basierend auf der maximalen Auftragslaufzeit zuzüglich einer Marge. Weitere Informationen zur Konfiguration des Amazon-S3-Lebenszyklus finden Sie in der Amazon-S3-Dokumentation unter Verwalten Ihres Speicherlebenszyklus.

- Eine IAM-Richtlinie mit entsprechenden Berechtigungen für Ihren Amazon S3-Pfad, die Sie Ihrer AWS Glue-Jobrolle zuordnen können.
- Wenn sich Ihre Vertica-Instance in einer Amazon VPC befindet, konfigurieren Sie Amazon VPC so, dass Ihr AWS Glue-Job mit der Vertica-Instance kommunizieren kann, ohne dass der Datenverkehr das öffentliche Internet durchquert.

  Identifizieren oder erstellen Sie in Amazon VPC eine VPC, ein Subnetz und eine Sicherheitsgruppe, die AWS Glue bei der Ausführung des Jobs verwendet. Darüber hinaus muss Amazon VPC so konfiguriert sein, dass der Netzwerkdatenverkehr zwischen Ihrer Vertica-Instance und diesem Standort zugelassen wird. Ihr Auftrag muss eine TCP-Verbindung mit Ihrem Vertica-Client-Port (Standard 5433) herstellen. Abhängig von Ihrem Netzwerklayout kann dies Änderungen an den Sicherheitsgruppenregeln, dem Netzwerk ACLs, den NAT-Gateways und den Peering-Verbindungen erfordern.

Sie können dann mit der Konfiguration von AWS Glue für die Verwendung mit Vertica fortfahren.

Eine Verbindung zu Vertica konfigurieren Sie wie folgt:

1. Erstellen Sie AWS Secrets Manager unter Verwendung Ihrer Vertica-Anmeldeinformationen einen geheimen Schlüssel, und`verticaUsername`. `verticaPassword` Um ein Geheimnis in Secrets Manager zu erstellen, folgen Sie dem Tutorial, das in der AWS Secrets Manager Dokumentation unter Create an AWS Secrets Manager Secret verfügbar ist. Nachdem Sie das

Geheimnis erstellt haben, behalten Sie den Geheimnamen *secretName* für den nächsten Schritt bei.

- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `user` mit dem Wert. *verticaUsername*
- Wenn Sie Schlüssel/Wert-Paare auswählen, erstellen Sie ein Paar für den Schlüssel `password` mit dem Wert. *verticaPassword*

2. Stellen Sie in der AWS Glue-Konsole eine Verbindung her, indem Sie den Schritten unter folgenthe section called “AWS Glue Verbindung hinzufügen”. Nachdem Sie die Verbindung hergestellt haben, behalten Sie den Verbindungsnamen für den nächsten Schritt bei. *connectionName*

   - Wählen Sie als Verbindungstyp Vertica aus.
   - Geben Sie als Vertica-Host den Hostnamen Ihrer Vertica-Installation an.
   - Geben Sie als Vertica-Port den Port an, über den Ihre Vertica-Installation verfügbar ist.
   - Geben Sie bei der Auswahl eines AWS Geheimnisses Folgendes an*secretName*.

3. In den folgenden Situationen ist möglicherweise eine zusätzliche Konfiguration erforderlich:

   - Für Vertica-Instances, die AWS in einer Amazon VPC gehostet werden
     - Stellen Sie Amazon VPC-Verbindungsinformationen für die AWS Glue-Verbindung bereit, die Ihre Vertica-Sicherheitsanmeldedaten definiert. Wenn Sie Ihre Verbindung erstellen oder aktualisieren, legen Sie VPC, Subnetz und Sicherheitsgruppen in den Netzwerkoptionen fest.

Nachdem Sie eine AWS Glue Vertica-Verbindung hergestellt haben, müssen Sie die folgenden Schritte ausführen, bevor Sie Ihre Verbindungsmethode aufrufen.

- Erteilen Sie die mit Ihrem AWS Glue-Job verknüpfte IAM-Rolle für. *tempS3Path*
- Erteilen Sie der mit Ihrem AWS Glue-Job verknüpften IAM-Rolle *secretName* Leserechte.
- Stellen *connectionName* Sie in Ihrer AWS Glue-Job-Konfiguration eine zusätzliche Netzwerkverbindung bereit.

Lesen aus Vertica

Voraussetzungen:

- Eine Vertica-Tabelle, aus der gelesen werden soll. Sie benötigen den Vertica-Datenbanknamen *dbName* und den Tabellennamen. *tableName*
- Eine AWS Glue Vertica-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu Vertica zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.
- Ein Amazon-S3-Bucket oder -Ordner, der (wie zuvor erwähnt) für den temporären Speicher verwendet werden soll. Sie benötigen den Namen,*tempS3Path*. Sie müssen über das `s3a`-Protokoll eine Verbindung zu diesem Standort herstellen.

Zum Beispiel:

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Sie können auch eine SELECT-SQL-Abfrage angeben, um die an Sie zurückgegebenen Ergebnisse zu filtern DynamicFrame oder um auf einen Datensatz aus mehreren Tabellen zuzugreifen.

Zum Beispiel:

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "query": "select * FROM tableName",
    },
)
```

Schreiben in Vertica-Tabellen

In diesem Beispiel werden Informationen aus einer vorhandenen Datei DynamicFrame *dynamicFrame* in Vertica geschrieben. Wenn die Tabelle bereits Informationen enthält, hängt AWS Glue Daten von Ihnen DynamicFrame an.

Voraussetzungen:

- Ein aktueller oder gewünschter Tabellenname*tableName*, in den Sie schreiben möchten. Sie benötigen außerdem den entsprechenden Vertica-Datenbanknamen,*dbName*.
- Eine AWS Glue Vertica-Verbindung, die für die Bereitstellung von Authentifizierungsinformationen konfiguriert ist. Gehen Sie wie im vorherigen Verfahren beschrieben vor, um eine Verbindung zu Vertica zu konfigurieren, um Ihre Authentifizierungsinformationen zu konfigurieren. Sie benötigen den Namen der AWS Glue-Verbindung,*connectionName*.
- Ein Amazon-S3-Bucket oder -Ordner, der (wie zuvor erwähnt) für den temporären Speicher verwendet werden soll. Sie benötigen den Namen,*tempS3Path*. Sie müssen über das `s3a`-Protokoll eine Verbindung zu diesem Standort herstellen.

Zum Beispiel:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Referenz zur Vertica-Verbindungsoption

- `connectionName` – Erforderlich. Wird für Lesen/Schreiben verwendet. Der Name einer AWS Glue Vertica-Verbindung, die so konfiguriert ist, dass sie Authentifizierungs- und Netzwerkinformationen für Ihre Verbindungsmethode bereitstellt.
- db – Erforderlich. Wird für Lesen/Schreiben verwendet. Der einer Datenbank in Vertica, mit dem Ihre Verbindungsmethode interagieren wird.

- `dbSchema` – Erforderlich, um ggf. Ihre Tabelle zu identifizieren. Wird für Lesen/Schreiben verwendet. Standard: `public`. Der Name eines Schemas, mit dem Ihre Verbindungsmethode interagieren wird.
- `table` – Zum Schreiben erforderlich, zum Lesen erforderlich, sofern `query` nicht angegeben ist. Wird für Lesen/Schreiben verwendet. Der Name einer Tabelle, mit der Ihre Verbindungsmethode interagieren wird.
- `query` – Wird zum Lesen verwendet. Eine SELECT-SQL-Abfrage, die definiert, was beim Lesen aus Teradata abgerufen werden soll.
- `staging_fs_url` – Erforderlich. Wird für Lesen/Schreiben verwendet. Gültige Werte:. `s3a` URLs Die URL eines Amazon-S3-Buckets oder -Ordners, der für den temporären Speicher verwendet werden soll.

DataFrame Optionen für ETL in AWS Glue 5.0 für Spark

A DataFrame ist ein Datensatz, der ähnlich einer Tabelle in benannte Spalten organisiert ist und Operationen im Funktionsstil (map/reduce/filter/etc.) und SQL-Operationen (Select, Project, Aggregate) unterstützt.

Um eine DataFrame für eine von Glue unterstützte Datenquelle zu erstellen, ist Folgendes erforderlich:

- Datenquellen-Konnektor `ClassName`
- Datenquellenverbindung `Options`

In ähnlicher Weise sind dieselben erforderlich, DataFrame um in eine von Glue unterstützte Datensenke zu schreiben:

- Anschluss für eine Datensenke `ClassName`
- Datensenkenverbindung `Options`

Beachten Sie, dass AWS Glue-Funktionen wie Job-Lesezeichen und DynamicFrame Optionen wie in nicht unterstützt `connectionName` werden DataFrame. Weitere Informationen zu DataFrame und den unterstützten Vorgängen finden Sie in der Spark-Dokumentation für DataFrame.

Den Konnektor angeben ClassName

Um die `ClassName` einer Datenquelle/Datensenke anzugeben, verwenden Sie die `.format` Option, um den entsprechenden Konnektor `ClassName` bereitzustellen, der die Datenquelle/Senke definiert.

JDBC-Konnektoren

Geben Sie für JDBC-Konnektoren den Wert der Option an und geben Sie `jdbc` den JDBC-Treiber in der `.format` Option an. `ClassName` `driver`

```
df = spark.read.format("jdbc").option("driver", "<DATA SOURCE JDBC DRIVER
 CLASSNAME>")...

df.write.format("jdbc").option("driver", "<DATA SINK JDBC DRIVER CLASSNAME>")...
```

In der folgenden Tabelle ist der JDBC-Treiber `ClassName` der unterstützten Datenquelle in AWS Glue for aufgeführt. DataFrames

Datenquelle	Treiber ClassName
PostgreSQL	org.postgresql.Driver
Oracle	oracle.jdbc.driver. OracleDriver
SQLServer	com.microsoft.sqlserver.jdbc. SQLServer Treiber
MySQL	com.mysql.jdbc.Driver
SAPHana	com.sap.db.jdbc.Treiber
Teradata	com.teradata.jdbc. TeraDriver

Spark-Steckverbinder

Geben Sie bei Funken-Steckverbindern den Wert `ClassName` des Steckers als Wert der `.format` Option an.

```
df = spark.read.format("<DATA SOURCE CONNECTOR CLASSNAME>")...
```

```
df.write.format("<DATA SINK CONNECTOR CLASSNAME>")...
```

In der folgenden Tabelle ist der Spark-Konnektor `ClassName` der unterstützten Datenquelle in AWS Glue for aufgeführt DataFrames.

Datenquelle	ClassName
MongoDB/DocumentDB	glue.spark.mongodb
Redshift	io.github.spark_redshift_community.spark.redshift
AzureCosmos	kosmos.oltp
AzureSQL	com.microsoft.sqlserver.jdbc.spark
BigQuery	com.google.cloud.spark.bigquery
OpenSearch	org.opensearch.spark.sql
Snowflake	net.snowflake.spark.snowflake
Vertica	com.vertica.spark.Datenquelle. VerticaSource

Angabe der Verbindungsoptionen

Um die `Options` Verbindung zu einer Datenquelle/Datensenke anzugeben, verwenden Sie die, `.option(<KEY>, <VALUE>)` um einzelne Optionen oder `.options(<MAP>)` mehrere Optionen als Key-Value-Map bereitzustellen.

Jede Datenquelle/Datensenke unterstützt ihren eigenen Verbindungssatz. `Options` Einzelheiten zu den verfügbaren `Options` Optionen finden Sie in der öffentlichen Dokumentation des jeweiligen Spark-Konnektors für Datenquelle/Datenquelle, der in der folgenden Tabelle aufgeführt ist.

- JDBC
- MongoDB/DocumentDB
- Redshift
- AzureCosmos
- AzureSQL

- BigQuery
- OpenSearch
- Snowflake
- Vertikal

Beispiele

Die folgenden Beispiele lesen aus PostgreSQL und schreiben hinein: SnowFlake

Python

Beispiel:

```
from awsglue.context import GlueContext
from pyspark.sql import SparkSession

spark = SparkSession.builder.getOrCreate()

dataSourceClassName = "jdbc"
dataSourceOptions = {
  "driver": "org.postgresql.Driver",
  "url": "<url>",
  "user": "<user>",
  "password": "<password>",
  "dbtable": "<dbtable>",
}

dataframe = spark.read.format(className).options(**options).load()

dataSinkClassName = "net.snowflake.spark.snowflake"
dataSinkOptions = {
  "sfUrl": "<url>",
  "sfUsername": "<username>",
  "sfPassword": "<password>",
  "sfDatabase" -> "<database>",
  "sfSchema" -> "<schema>",
  "sfWarehouse" -> "<warehouse>"
}

dataframe.write.format(dataSinkClassName).options(**dataSinkOptions).save()
```

Scala

Beispiel:

```
import org.apache.spark.sql.SparkSession

val spark = SparkSession.builder().getOrCreate()

val dataSourceClassName = "jdbc"
val dataSourceOptions = Map(
  "driver" -> "org.postgresql.Driver",
  "url" -> "<url>",
  "user" -> "<user>",
  "password" -> "<password>",
  "dbtable" -> "<dbtable>"
)

val dataframe =
 spark.read.format(dataSourceClassName).options(dataSourceOptions).load()

val dataSinkClassName = "net.snowflake.spark.snowflake"
val dataSinkOptions = Map(
  "sfUrl" -> "<url>",
  "sfUsername" -> "<username>",
  "sfPassword" -> "<password>",
  "sfDatabase" -> "<database>",
  "sfSchema" -> "<schema>",
  "sfWarehouse" -> "<warehouse>"
)

dataframe.write.format(dataSinkClassName).options(dataSinkOptions).save()
```

Custom- und AWS Marketplace ConnectionType-Werte

Diese umfassen u. a. folgende:

- `"connectionType": "marketplace.athena"`: Bezeichnet eine Verbindung zu einem Amazon-Athena-Datenspeicher. Die Verbindung verwendet einen Konnektor von AWS Marketplace.
- `"connectionType": "marketplace.spark"`: Bezeichnet eine Verbindung zu einem Apache-Spark-Datenspeicher. Die Verbindung verwendet einen Anschluss von AWS Marketplace.
- `"connectionType": "marketplace.jdbc"`: Bezeichnet eine Verbindung zu einem JDBC-Datenspeicher. Die Verbindung verwendet einen Anschluss von AWS Marketplace.

- `"connectionType": "custom.athena"`: Bezeichnet eine Verbindung zu einem Amazon-Athena-Datenspeicher. Die Verbindung verwendet einen benutzerdefinierten Connector, auf den Sie hochladen AWS Glue Studio.
- `"connectionType": "custom.spark"`: Bezeichnet eine Verbindung zu einem Apache-Spark-Datenspeicher. Die Verbindung verwendet einen benutzerdefinierten Connector, auf den Sie hochladen AWS Glue Studio.
- `"connectionType": "custom.jdbc"`: Bezeichnet eine Verbindung zu einem JDBC-Datenspeicher. Die Verbindung verwendet einen benutzerdefinierten Connector, auf den Sie hochladen AWS Glue Studio.

Verbindungsoptionen für den Typ custom.jdbc oder marketplace.jdbc

- `className` – Zeichenfolge, erforderlich, Name der Treiberklasse.
- `connectionName` – Zeichenfolge, erforderlich, Name der Verbindung, die dem Konnektor zugeordnet ist.
- `url` – Zeichenfolge, erforderlich, JDBC-URL mit Platzhaltern (`${}`), die verwendet werden, um die Verbindung zur Datenquelle herzustellen. Der Platzhalter `${secretKey}` wird durch das Secret des gleichen Namens in AWS Secrets Manager ersetzt. Weitere Informationen zum Erstellen der URL finden Sie in der Dokumentation zum Datenspeicher.
- `secretId` oder `user/password` – Zeichenfolge, erforderlich, zum Abrufen der Anmeldeinformationen für die URL.
- `dbTable` oder `query` – Zeichenfolge, erforderlich, die Tabelle oder SQL-Abfrage, aus der die Daten abgerufen werden. Sie können `dbTable` oder `query` angeben, aber nicht beides.
- `partitionColumn` – Zeichenfolge, optional, der Name einer Ganzzahlspalte, die für die Partitionierung verwendet wird. Diese Option funktioniert nur, wenn sie in `lowerBound`, `upperBound` und `numPartitions` enthalten ist. Diese Option funktioniert auf die gleiche Weise wie im Spark SQL JDBC Reader. Weitere Informationen finden Sie unter JDBC To Other Databases im Apache Spark SQL DataFrames and Datasets Guide.

  Die Werte für `lowerBound` und `upperBound` werden verwendet, um den Partitionsschritt zu bestimmen, nicht zum Filtern der Zeilen in der Tabelle. Alle Zeilen der Tabelle werden partitioniert und zurückgegeben.

> **Note**
>
> Wenn Sie eine Abfrage anstelle eines Tabellennamens verwenden, sollten Sie überprüfen, ob die Abfrage mit der angegebenen Partitionierungsbedingung funktioniert. Zum Beispiel:
>
> - Wenn Ihr Abfrageformat `"SELECT col1 FROM table1"` lautet, dann testen Sie die Abfrage, indem Sie eine `WHERE`-Klausel am Ende der Abfrage stellen, die die Partitionsspalte verwendet.
> - Wenn Ihr Abfrageformat `SELECT col1 FROM table1 WHERE col2=val"` lautet, dann testen Sie die Abfrage, indem Sie die `WHERE`-Klausel mit `AND` und einem Ausdruck erweitern, der die Partitionsspalte verwendet.

- `lowerBound` – Ganzzahl, optional, der Mindestwert von `partitionColumn`, der verwendet wird, um Partitionsschritte festzulegen.
- `upperBound` – Ganzzahl, optional, der Maximalwert von `partitionColumn`, der verwendet wird, um Partitionsschritte festzulegen.
- `numPartitions` – Ganzzahl, optional, die Anzahl der Partitionen. Dieser Wert, zusammen mit `lowerBound` (inklusive) und `upperBound` (exklusiv), bilden Partitionsschritte für generierte `WHERE`-Klauselausdrücke, die verwendet werden, um die `partitionColumn` aufzuteilen.

> **Important**
>
> Seien Sie vorsichtig mit der Anzahl der Partitionen, da zu viele Partitionen Probleme auf Ihren externen Datenbanksystemen verursachen können.

- `filterPredicate` – Zeichenfolge, optional, zusätzliche Bedingungsklausel zum Filtern von Daten aus der Quelle. Zum Beispiel:

```
BillingCity='Mountain View'
```

  Wenn Sie eine Abfrage anstelle eines Tabellennamens verwenden, sollten Sie überprüfen, ob die Abfrage mit dem angegebenen `filterPredicate` funktioniert. Zum Beispiel:

  - Wenn Ihr Abfrageformat `"SELECT col1 FROM table1"` lautet, dann testen Sie die Abfrage, indem Sie eine `WHERE`-Klausel am Ende der Abfrage stellen, die das Filterprädikat verwendet.

  - Wenn Ihr Abfrageformat `"SELECT col1 FROM table1 WHERE col2=val"` lautet, dann testen Sie die Abfrage, indem Sie die `WHERE`-Klausel mit `AND` und einem Ausdruck erweitern, der das Filterprädikat verwendet.
- `dataTypeMapping` – Wörterbuch, optional, benutzerdefiniertes Datentyp-Mapping, das ein Mapping aus einem JDBC-Datentyp auf einen Glue-Datentyp durchführt. Die Option `"dataTypeMapping":{"FLOAT":"STRING"}` ordnet beispielsweise Datenfelder vom Typ JDBC dem `String` Typ Java zu, indem `FLOAT` sie die `ResultSet.getString()` Methode des Treibers aufruft, und verwendet sie zum Erstellen AWS Glue Aufzeichnungen. Das `ResultSet`-Objekt wird von jedem Treiber implementiert, sodass das Verhalten spezifisch für den von Ihnen verwendeten Treiber ist. Informieren Sie sich in der Dokumentation für Ihren JDBC-Treiber, um zu verstehen, wie der Treiber die Konvertierungen durchführt.
- Das Tool AWS Glue Derzeit werden folgende Datentypen unterstützt:
  - DATUM
  - STRING
  - TIMESTAMP (ZEITSTEMPEL)
  - INT
  - FLOAT
  - LONG
  - BIGDECIMAL
  - BYTE
  - SHORT
  - DOUBLE

  Die unterstützten JDBC-Datentypen sind Java8 java.sql.types.

  Die standardmäßigen Datentypzuordnungen (von JDBC zu AWS Glue) sind:

  - DATUM -> DATUM
  - VARCHAR -> ZEICHENFOLGE
  - CHAR -> ZEICHENFOLGE
  - LONGNVARCHAR -> ZEICHENFOLGE
  - TIMESTAMP -> ZEITSTEMPEL
  - INTEGER -> INT
  - FLOAT -> FLOAT

- REAL -> FLOAT
- BIT -> BOOLESCHER WERT
- BOOLEAN -> BOOLESCHER WERT
- BIGINT -> LANG
- DECIMAL -> BIGDECIMAL
- NUMERIC -> BIGDECIMAL
- TINYINT -> KURZ
- SMALLINT -> KURZ
- DOUBLE -> DOPPELT

Wenn Sie ein benutzerdefiniertes Datentyp-Mapping mit der Option `dataTypeMapping` verwenden, können Sie ein Standard-Datentyp-Mapping überschreiben. Nur die JDBC-Datentypen, die in der Option `dataTypeMapping` betroffen sind. Das Standardmapping wird für alle anderen JDBC-Datentypen verwendet. Sie können bei Bedarf Mappings für zusätzliche JDBC-Datentypen hinzufügen. Wenn ein JDBC-Datentyp weder in der Standardzuordnung noch in einer benutzerdefinierten Zuordnung enthalten ist, wird der Datentyp in den AWS Glue `STRING`Datentyp standardmäßig.

Das folgende Python-Codebeispiel zeigt, wie mit AWS Marketplace JDBC-Treibern aus JDBC-Datenbanken gelesen wird. Es demonstriert das Lesen aus einer Datenbank und das Schreiben in einen S3-Speicherort.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

```
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.jdbc", connection_options =
     {"dataTypeMapping":{"INTEGER":"STRING"},"upperBound":"200","query":"select id,
       name, department from department where id < 200","numPartitions":"4",
       "partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-
jdbc"},
        transformation_ctx = "DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"},
      "upperBound":"200","query":"select id, name, department from department where
       id < 200","numPartitions":"4","partitionColumn":"id","lowerBound":"0",
       "connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
    ## @type: ApplyMapping
    ## @args: [mappings = [("department", "string", "department", "string"), ("name",
 "string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
 "Transform0"]
    ## @return: Transform0
    ## @inputs: [frame = DataSource0]
    Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
 "string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
 "id", "int")],
      transformation_ctx = "Transform0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = Transform0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
      connection_type = "s3", format = "json", connection_options = {"path":
      "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
    job.commit()
```

Verbindungsoptionen für den Typ custom.athena oder marketplace.athena

- `className` – Zeichenfolge, erforderlich, Name der Treiberklasse. Wenn Sie den CloudWatch Athena-Konnektor verwenden, ist dieser Parameterwert das Präfix des Klassennamens (z. B.`"com.amazonaws.athena.connectors"`). Der CloudWatch Athena-Connector besteht aus zwei Klassen: einem Metadaten-Handler und einem Record-Handler. Wenn Sie hier das allgemeine Präfix angeben, lädt die API die richtigen Klassen basierend auf diesem Präfix.

- `tableName`— Zeichenfolge, erforderlich, der Name des zu CloudWatch lesenden Log-Streams. Dieses Codefragment verwendet den speziellen Ansichtsnamen `all_log_streams`, was bedeutet, dass der zurückgegebene dynamische Datenrahmen Daten aus allen Protokollstreams in der Protokollgruppe enthält.
- `schemaName`— Zeichenfolge, erforderlich, der Name der CloudWatch Protokollgruppe, aus der gelesen werden soll. Beispiel, `/aws-glue/jobs/output`.
- `connectionName` – Zeichenfolge, erforderlich, Name der Verbindung, die dem Konnektor zugeordnet ist.

Weitere Optionen für diesen Connector finden Sie in der Amazon Athena CloudWatch Connector-README-Datei unter. GitHub

Im folgenden Python-Codebeispiel wird gezeigt, wie aus einem Athena-Datenspeicher mithilfe eines AWS Marketplace -Konnektoren gelesen wird. Es demonstriert das Lesen aus Athena und das Schreiben in einen S3-Speicherort.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [connection_type = "marketplace.athena", connection_options =
 {"tableName":"all_log_streams","schemaName":"/aws-glue/jobs/output",
  "connectionName":"test-connection-athena"}, transformation_ctx = "DataSource0"]
## @return: DataSource0
## @inputs: []
DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
  "marketplace.athena", connection_options = {"tableName":"all_log_streams",,
  "schemaName":"/aws-glue/jobs/output","connectionName":
```

```
      "test-connection-athena"}, transformation_ctx = "DataSource0")
   ## @type: ApplyMapping
   ## @args: [mappings = [("department", "string", "department", "string"), ("name",
"string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
"Transform0"]
   ## @return: Transform0
   ## @inputs: [frame = DataSource0]
   Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
"string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
"id", "int")],
      transformation_ctx = "Transform0")
   ## @type: DataSink
   ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
    "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
   ## @return: DataSink0
   ## @inputs: [frame = Transform0]
   DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
     connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
   job.commit()
```

Verbindungsoptionen für den Typ custom.spark oder marketplace.spark

- `className` – Zeichenfolge, erforderlich, Konnektor-Klassenname.
- `secretId` – Zeichenfolge, optional, wird zum Abrufen der Anmeldeinformationen für die Konnektor-Verbindung verwendet.
- `connectionName` – Zeichenfolge, erforderlich, Name der Verbindung, die dem Konnektor zugeordnet ist.
- Andere Optionen hängen vom Datenspeicher ab. Beispielsweise beginnen die OpenSearch Konfigurationsoptionen mit dem Präfixes, wie in der Elasticsearch for Apache Hadoop-Dokumentation beschrieben. Spark-Verbindungen zu Snowflake verwenden Optionen wie `sfUser` und `sfPassword`, wie unter Using the Spark Connector (Verwenden des Spark-Connectors) im Handbuch Connecting to Snowflake (Verbindung mit Snowflake herstellen) beschrieben.

Das folgende Python-Codebeispiel zeigt, wie mithilfe einer `marketplace.spark` Verbindung aus einem OpenSearch Datenspeicher gelesen wird.

```
   import sys
```

```
    from awsglue.transforms import *
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])

    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.spark", connection_options =
{"path":"test",
     "es.nodes.wan.only":"true","es.nodes":"https://<AWS endpoint>",
     "connectionName":"test-spark-es","es.port":"443"}, transformation_ctx =
"DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.spark", connection_options = {"path":"test","es.nodes.wan.only":
      "true","es.nodes":"https://<AWS endpoint>","connectionName":
      "test-spark-es","es.port":"443"}, transformation_ctx = "DataSource0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
         "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = DataSource0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = DataSource0,
       connection_type = "s3", format = "json", connection_options = {"path":
       "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
    job.commit()
```

Allgemeine Optionen

Die Optionen in diesem Abschnitt werden als `connection_options` bereitgestellt, gelten jedoch nicht speziell für einen Konnektor.

Die folgenden Parameter werden im Allgemeinen bei der Konfiguration von Lesezeichen verwendet. Sie können für Amazon-S3- oder JDBC-Workflows gelten. Weitere Informationen finden Sie unter the section called “Verwenden von Auftragslesezeichen”.

- `jobBookmarkKeys` – Ein Array von Spaltennamen.
- `jobBookmarkKeysSortOrder` – Zeichenfolge, die definiert, wie Werte basierend auf der Sortierreihenfolge verglichen werden. Zulässige Werte: `"asc"`, `"desc"`.
- `useS3ListImplementation` – Wird zur Verwaltung der Speicherleistung beim Auflisten von Amazon-S3-Bucket-Inhalten verwendet. Weitere Informationen finden Sie unter Optimieren der Speicherverwaltung in AWS Glue.

Datenformatoptionen für Eingaben und Ausgaben in AWS Glue für Spark

Diese Seiten bieten Informationen zur Feature-Unterstützung und Konfigurationsparameter für Datenformate, die von AWS Glue für Spark unterstützt werden. Im Folgenden finden Sie eine Beschreibung der Verwendung und Anwendbarkeit dieser Informationen.

Funktionsunterstützung für alle Datenformate in AWS Glue

Jedes Datenformat kann unterschiedliche AWS Glue-Funktionen unterstützen. Die folgenden allgemeinen Features werden je nach Formattyp möglicherweise nicht unterstützt. Informieren Sie sich in der Dokumentation für Ihr Datenformat, um zu verstehen, wie Sie unsere Features zur Erfüllung Ihrer Anforderungen nutzen können.

Lesen	AWS Glue kann dieses Datenformat ohne zusätzliche Ressourcen wie Konnektoren erkennen und interpretieren.
Schreibe	AWS Glue kann Daten in diesem Format ohne zusätzliche Ressourcen schreiben. Sie können Bibliotheken von Drittanbietern in Ihren Job einbeziehen und Standardfunktionen von Apache Spark verwenden, um Daten wie in anderen Spark-Umgebungen zu schreiben. Weitere Informationen einschließlich Bibliothe ken finden Sie unter the section called “Python-Bibliotheken”.

Streamin gelesen	AWS Glue kann dieses Datenformat aus einem Apache Kafka-, Amazon Managed Streaming for Apache Kafka- oder Amazon Kinesis Kinesis-Nachrichtenstream erkennen und interpretieren. Wir erwarten, dass Streams Daten in einem konsistenten Format präsentie ren, sodass sie als `DataFrames` eingelesen werden.
Gruppier n von kleinen Dateien	AWS Glue kann Dateien gruppieren, um Batch-Arbeit zu erstellen, die bei der Durchführ ung von AWS Glue-Transformationen an jeden Knoten gesendet wird. Dies kann die Leistung für Workloads mit großen Mengen kleiner Dateien erheblich verbessern. Weitere Informationen finden Sie unter the section called “Gruppieren von Eingabedateien”.
Auftrags-Lesezeic en	AWS Glue kann mithilfe von Job-Lesezeichen den Fortschritt von Transformationen verfolgen , die dieselbe Arbeit an demselben Datensatz über Jobläufe hinweg ausführen. Dies kann die Leistung für Workloads mit Datensätz en verbessern, bei denen seit der letzten Auftragsausführung nur an neuen Daten gearbeitet werden muss. Weitere Informationen finden Sie unter the section called “Verfolgen von verarbeiteten Daten mit Auftragslesezeiche n”.

Parameter, die für die Interaktion mit Datenformaten in AWS Glue verwendet werden

Bestimmte AWS Glue-Verbindungstypen unterstützen mehrere `format` Typen, sodass Sie Informationen über Ihr Datenformat mit einem `format_options` Objekt angeben müssen, wenn Sie Methoden wie verwenden`GlueContext.write_dynamic_frame.from_options`.

- `s3`— Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter. Sie können auch die Dokumentation für die Methoden anzeigen, die diesen Verbindungstyp unterstützt: the section called “create_dynamic_frame_from_options” und the section called “write_dynamic_frame_from_options” in Python sowie die entsprechende Scala-Methoden the section called “getSourceWithFormatieren” und the section called “getSinkWithFormatieren”.

- `kinesis`— Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:Kinesis-Verbindungsparameter. Sie können auch die Dokumentation für die Methode anzeigen, die diesen Verbindungstyp unterstützt: the section called “create_data_frame_from_options” und die entsprechende Scala-Methode the section called “createDataFrameFromOptions”.
- `kafka`— Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:Kafka-Verbindungsparameter. Sie können auch die Dokumentation für die Methode anzeigen, die diesen Verbindungstyp unterstützt: the section called “create_data_frame_from_options” und die entsprechende Scala-Methode the section called “createDataFrameFromOptions”.

Einige Verbindungstypen erfordern `format_options` nicht. Im Verlauf der normalen Verwendung ruft eine JDBC-Verbindung zu einer relationalen Datenbank zum Beispiel Daten in einem konsistenten, tabellarischen Datenformat ab. Daher wäre für das Lesen von einer JDBC-Verbindung `format_options` nicht erforderlich.

Für einige Methoden zum Lesen und Schreiben von Daten in Glue ist `format_options` nicht erforderlich. Zum Beispiel `GlueContext.create_dynamic_frame.from_catalog` mit AWS Glue-Crawlern. Crawler bestimmen die Form Ihrer Daten. Wenn Sie Crawler verwenden, untersucht ein AWS Glue-Klassifikator Ihre Daten, um fundierte Entscheidungen darüber zu treffen, wie Ihr Datenformat dargestellt werden soll. Anschließend wird eine Darstellung Ihrer Daten im Glue-Datenkatalog gespeichert, die in einem AWS AWS Glue-ETL-Skript verwendet werden kann, um Ihre Daten mit der `GlueContext.create_dynamic_frame.from_catalog` Methode abzurufen. Crawler machen es überflüssig, Informationen über Ihr Datenformat manuell anzugeben.

Für Jobs, die auf AWS Lake Formation verwaltete Tabellen zugreifen, AWS Glue unterstützt das Lesen und Schreiben aller Formate, die von Lake Formation verwalteten Tabellen unterstützt werden. Die aktuelle Liste der unterstützten Formate für gesteuerte AWS Lake Formation Tabellen finden Sie unter Hinweise und Einschränkungen für verwaltete Tabellen im AWS Lake Formation Entwicklerhandbuch.

> **Note**
>
> Um Apache Parquet zu schreiben, AWS Glue ETL unterstützt nur das Schreiben in eine kontrollierte Tabelle, wenn eine Option für einen benutzerdefinierten Parquet-Writer-Typ angegeben wird, der für dynamische Frames optimiert ist. Beim Schreiben an eine verwaltete Tabelle mit dem `parquet`-Format sollten Sie den Schlüssel `useGlueParquetWriter` mit einem Wert von `true` den Tabellenparametern hinzufügen.

Themen

- Verwenden des CSV-Formats in AWS Glue
- Das Parkett-Format in AWS Glue verwenden
- Verwenden des XML-Formats in AWS Glue
- Verwenden des Avro-Formats in Glue AWS
- Verwenden des GrokLog-Formats in Glue AWS
- Das Ion-Format in AWS Glue verwenden
- Verwenden des JSON-Formats in AWS Glue
- Verwenden des ORC-Formats in Glue AWS
- Verwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs
- Freigegebene Konfigurationsreferenz

Verwenden des CSV-Formats in AWS Glue

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im CSV-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des CSV-Formats (Comma-Separated Value). Dieses Format ist ein minimales, zeilenbasiertes Datenformat. CSVs entsprechen oft nicht strikt einem Standard, aber weitere Informationen finden Sie in RFC 4180 und RFC 7111.

Sie können AWS Glue verwenden, um CSVs aus Amazon S3 und aus Streaming-Quellen zu lesen sowie in Amazon S3 CSVs zu schreiben. Sie können `bzip`- und `gzip`-Archive mit CSV-

Dateien aus S3 lesen und schreiben. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen AWS Glue-Funktionen die CSV-Formatoption unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Unterstützt	Unterstützt	Unterstützt	Unterstützt

Beispiel: Lesen von CSV-Dateien oder Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den CSV-Dateien oder Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="csv"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben. Sie können konfigurieren, wie der Reader mit S3 in `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter. Sie können konfigurieren, wie der Reader CSV-Dateien in Ihrem `format_options` interpretiert. Einzelheiten finden Sie unter CSV-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Lesens von CSV-Dateien oder -Ordnern aus S3.

Wir bieten einen benutzerdefinierten CSV-Reader mit Leistungsoptimierungen für gängige Workflows über den `optimizePerformance`-Konfigurationsschlüssel. Um festzustellen, ob dieser Reader für Ihren Workload geeignet ist, lesen Sie the section called “Verwendung eines optimierten CSV-Reader”.

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
# Example: Read CSV from S3
# For show, we handle a CSV with a header row.  Set the withHeader option.
# Consider whether optimizePerformance is right for your workflow.

from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="csv",
    format_options={
        "withHeader": True,
        # "optimizePerformance": True,
    },
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame = spark.read\
    .format("csv")\
    .option("header", "true")\
    .load("s3://s3path")
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
// Example: Read CSV from S3
// For show, we handle a CSV with a header row.  Set the withHeader option.
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"withHeader": true}"""),
      connectionType="s3",
```

```
      format="csv",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Sie können es auch DataFrames in einem Skript (`org.apache.spark.sql.DataFrame`) verwenden.

```
val dataFrame = spark.read
  .option("header","true")
  .format("csv")
  .load("s3://s3path“)
```

Beispiel: Schreiben von CSV-Dateien und -Ordnern in S3

Voraussetzungen: Sie benötigen ein initialisiertes DataFrame (`dataFrame`) oder ein DynamicFrame (`dynamicFrame`). Sie benötigen auch Ihren erwarteten S3-Ausgabepfad, `s3path`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="csv"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben. Sie können konfigurieren, wie der Writer mit S3 in `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter. Sie können konfigurieren, wie Ihre Operation den Inhalt Ihrer Dateien in `format_options` schreibt. Einzelheiten finden Sie unter CSV-Konfigurationsreferenz. Das folgende AWS -Glue-ETL-Skript zeigt den Prozess des Schreibens von CSV-Dateien und -Ordnern in S3.

Python

Verwenden Sie für dieses Beispiel die Methode write_dynamic_frame.from_options.

```
# Example: Write CSV to S3
# For show, customize how we write string type values.  Set quoteChar to -1 so our
 values are not quoted.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
```

```
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="csv",
    format_options={
        "quoteChar": -1,
    },
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame.write\
    .format("csv")\
    .option("quote", None)\
    .mode("append")\
    .save("s3://s3path")
```

Scala

Verwenden Sie für dieses Beispiel die getSinkWithFormat-Methode.

```
// Example: Write CSV to S3
// For show, customize how we write string type values. Set quoteChar to -1 so our
 values are not quoted.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="csv"
    ).writeDynamicFrame(dynamicFrame)
```

```
  }
}
```

Sie können auch DataFrames in einem Skript (`org.apache.spark.sql.DataFrame`) verwenden.

```
dataFrame.write
    .format("csv")
    .option("quote", null)
    .mode("Append")
    .save("s3://s3path")
```

CSV-Konfigurationsreferenz

Sie können Folgendes verwenden, `format_options` wo immer die AWS Glue-Bibliotheken dies angeben`format="csv"`:

- `separator` – Gibt das Trennzeichen an. Der Standardwert ist ein Komma, aber es kann jedes andere Zeichen angegeben werden.
  - Typ: Text, Standard: `","`
- `escaper` – Gibt ein Zeichen an, das für die Maskierung verwendet werden soll. Diese Option wird nur beim Lesen von CSV-Dateien verwendet, nicht beim Schreiben. Wenn aktiviert, wird das unmittelbar folgende Zeichen unverändert verwendet, mit Ausnahme einiger bekannter Maskierungen (`\n`, `\r`, `\t` und `\0`).
  - Typ: Text, Standard: keine
- `quoteChar` – Gibt das Zeichen für Anführungszeichen an. Der Standardwert ist ein doppeltes Anführungszeichen. Setzen Sie dies auf `-1`, um Anführungszeichen generell zu deaktivieren.
  - Typ: Text, Standard: `'"'`
- `multiLine` – Gibt an, ob ein einzelner Datensatz mehrere Zeilen umfassen kann. Dies kommt vor, wenn ein Feld ein Neue-Zeile-Zeichen in Anführungszeichen enthält. Sie müssen diese Option auf "`True` setzen, wenn ein Datensatz mehrere Zeilen umfasst. Das Aktivieren von `multiLine` kann die Leistung verringern, da beim Parsen eine vorsichtigere Dateiaufteilung erforderlich ist.
  - Typ: Boolesch, Standard: `false`
- `withHeader` – Gibt an, ob die erste Zeile als Kopfzeile behandelt werden soll. Diese Option kann in der `DynamicFrameReader`-Klasse verwendet werden.
  - Typ: Boolesch, Standard: `false`

- `writeHeader` – Gibt an, ob die Kopfzeile ausgegeben werden soll. Diese Option kann in der `DynamicFrameWriter`-Klasse verwendet werden.
  - Typ: Boolesch, Standard: `true`
- `skipFirst` – Gibt an, ob die erste Datenzeile übersprungen werden soll.
  - Typ: Boolesch, Standard: `false`
- `optimizePerformance` – Gibt an, ob der erweiterte SIMD-CSV-Reader zusammen mit Apache-Arrow-basierten Spaltenspeicherformaten verwendet werden soll. Nur verfügbar in AWS Glue 3.0+.
  - Typ: Boolesch, Standard: `false`
- `strictCheckForQuoting`— Beim Schreiben kann Glue Werte CSVs, die es als Zeichenketten interpretiert, in Anführungszeichen setzen. Dies geschieht, um Unklarheiten beim Ausschreiben zu vermeiden. Um bei der Entscheidung, was geschrieben werden soll, Zeit zu sparen, zitiert Glue möglicherweise in bestimmten Situationen, in denen Anführungszeichen nicht erforderlich sind. Durch die Aktivierung einer strengen Prüfung wird eine intensivere Berechnung durchgeführt und es werden nur Angebote erstellt, wenn dies unbedingt erforderlich ist. Nur verfügbar in AWS Glue 3.0+.
  - Typ: Boolesch, Standard: `false`

Optimieren der Leseleistung mit vektorisiertem SIMD-CSV-Leser

AWS Glue Version 3.0 fügt einen optimierten CSV-Reader hinzu, der die allgemeine Arbeitsleistung im Vergleich zu zeilenbasierten CSV-Readern erheblich beschleunigen kann.

Der optimierte Reader:

- Verwendet CPU-SIMD-Anweisungen zum Lesen von der Festplatte
- Schreibt sofort Datensätze in einem Spaltenformat in den Speicher (Apache Arrow)
- Teilt die Datensätze in Batches auf

Dies spart Verarbeitungszeit, wenn die Datensätze später in Batches oder in ein Spaltenformat konvertiert werden sollen. Einige Beispiele sind das Ändern von Schemas oder das Abrufen von Daten nach Spalte.

Um den optimierten Reader zu verwenden, setzen Sie `"optimizePerformance"` auf `true` in der `format_options`- oder Tabelleneigenschaft.

```
glueContext.create_dynamic_frame.from_options(
```

```
    frame = datasource1,
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path"]},
    format = "csv",
    format_options={
        "optimizePerformance": True,
        "separator": ","
        },
    transformation_ctx = "datasink2")
```

Einschränkungen für den vektorisierten CSV-Reader

Beachten Sie die folgenden Einschränkungen des vektorisierten CSV-Readers:

- Die Formatoptionen `multiLine` und `escaper` werden nicht unterstützt. Es wird der Standard `escaper` doppelter Anführungszeichen `'"'` verwendet. Wenn diese Optionen festgelegt sind, AWS Glue greift automatisch auf die Verwendung des zeilenbasierten CSV-Readers zurück.
- Das Erstellen eines mit wird nicht unterstützt. DynamicFrame ChoiceType
- Das Erstellen von Datensätzen DynamicFrame mit Fehlern wird nicht unterstützt.
- Das Lesen von CSV-Dateien mit Multibyte-Zeichen wie japanischen oder chinesischen Zeichen wird nicht unterstützt.

Das Parkett-Format in AWS Glue verwenden

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im Parquet-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des Parquet-Formats. Dieses Format ist ein leistungsorientiertes, spaltenbasiertes Datenformat. Eine Einführung in das Format durch die Standardautorität finden Sie unter Dokumentationsübersicht zu Apache-Parquet.

Sie können AWS Glue verwenden, um Parquet-Dateien von Amazon S3 und aus Streaming-Quellen zu lesen sowie Parquet-Dateien auf Amazon S3 zu schreiben. Sie können `bzip`- und `gzip`-Archive mit Parquet-Dateien aus S3 lesen und schreiben. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen Funktionen von AWS Glue die Formatoption Parquet unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Unterstützt	Unterstützt	Nicht unterstützt	Unterstützt*

* Wird in AWS Glue Version 1.0+ unterstützt

Beispiel: Lesen von Parquet-Dateien oder -Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den Parquet-Dateien oder -Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="parquet"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um Ihren `s3path` anzugeben.

Sie können konfigurieren, wie der Reader mit S3 in der `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter.

Sie können konfigurieren, wie der Reader Parquet-Dateien in Ihrem `format_options` interpretiert. Details dazu finden Sie unter Parquet-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Lesens von Parquet-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
# Example: Read Parquet from S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path/"]},
```

```
    format = "parquet"
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame = spark.read.parquet("s3://s3path/")
```

Scala

Verwenden Sie für dieses Beispiel die getSourceWithFormat-Methode.

```
// Example: Read Parquet from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="parquet",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Sie können es auch DataFrames in einem Skript (`org.apache.spark.sql.DataFrame`) verwenden.

```
spark.read.parquet("s3://s3path/")
```

Beispiel: Schreiben von Parquet-Dateien und -Ordnern nach S3

Voraussetzungen: Sie benötigen ein initialisiertes DataFrame (`dataFrame`) oder DynamicFrame (`dynamicFrame`). Sie benötigen auch Ihren erwarteten S3-Ausgabepfad, `s3path`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="parquet"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben.

Sie können die Art und Weise, wie der Writer mit S3 in `connection_options` interagiert, weiter verändern. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter. Sie können konfigurieren, wie Ihre Operation den Inhalt Ihrer Dateien in `format_options` schreibt. Details dazu finden Sie unter Parquet-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Schreibens von Parquet-Dateien und -Ordnern nach S3.

Wir bieten über den `useGlueParquetWriter` Konfigurationsschlüssel einen benutzerdefinierten Parquet-Writer mit Leistungsoptimierungen für DynamicFrames. Um festzustellen, ob dieser Writer für Ihren Workload geeignet ist, lesen Sie Glue-Parquet-Writer.

Python

Verwenden Sie für dieses Beispiel die Methode write_dynamic_frame.from_options.

```
# Example: Write Parquet to S3
# Consider whether useGlueParquetWriter is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="parquet",
    connection_options={
        "path": "s3://s3path",
    },
    format_options={
        # "useGlueParquetWriter": True,
    },
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
df.write.parquet("s3://s3path/")
```

Scala

Verwenden Sie für dieses Beispiel die getSinkWithFormat-Methode.

```
// Example: Write Parquet to S3
// Consider whether useGlueParquetWriter is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="parquet"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Sie können es auch DataFrames in einem Skript (`org.apache.spark.sql.DataFrame`) verwenden.

```
df.write.parquet("s3://s3path/")
```

Parquet-Konfigurationsreferenz

Sie können Folgendes verwenden, `format_options` wo immer die AWS Glue-Bibliotheken dies angeben`format="parquet"`:

- `useGlueParquetWriter`— Gibt die Verwendung eines benutzerdefinierten Parquet-Writers an, der Leistungsoptimierungen für DynamicFrame Workflows bietet. Einzelheiten zur Verwendung finden Sie unter Glue-Parquet-Writer.
  - Typ: Boolesch, Standard:`false`

- `compression` – Gibt den verwendeten Komprimierungs-Codec an. Werte sind voll kompatibel mit `org.apache.parquet.hadoop.metadata.CompressionCodecName`.
  - Typ: Aufzählungstext, Standard: `"snappy"`
  - Werte: `"uncompressed"`, `"snappy"`, `"gzip"` und `"lzo"`
- `blockSize` – Gibt die Größe einer im Arbeitsspeicher gepufferten Zeilengruppe in Bytes an. Sie verwenden dies, um die Leistung zu optimieren. Die Größe sollte sich genau in eine Anzahl von Megabyte teilen.
  - Typ: Numerisch, Standard:`134217728`
  - Der Standardwert ist gleich 128 MB.
- `pageSize` – Gibt die Größe einer Seite in Byte an. Sie verwenden dies, um die Leistung zu optimieren. Eine Seite ist die kleinste Einheit, die vollständig gelesen werden muss, um auf einen einzelnen Datensatz zugreifen zu können.
  - Typ: Numerisch, Standard:`1048576`
  - Der Standardwert ist gleich 1 MB.

> **Note**
>
> Es können zusätzlich alle Optionen, die vom zugrunde liegenden SparkSQL-Code akzeptiert werden, im Rahmen des `connection_options`-Map-Parameters in dieses Format übergeben werden. Sie können beispielsweise eine Spark-Konfiguration wie MergeSchema für den einrichten AWS Glue Spark-Reader, um das Schema für alle Dateien zusammenzuführen.

Optimieren der Schreibleistung mit AWS -Glue-Parquet-Writer

> **Note**
>
> Der AWS Glue Parquet-Writer wurde in der Vergangenheit über den `glueparquet` Formattyp aufgerufen. Dieses Zugriffsmuster wird nicht mehr befürwortet. Verwenden Sie stattdessen den `parquet`-Typ mit aktiviertem `useGlueParquetWriter`.

Der AWS Glue Parquet-Writer verfügt über Leistungsverbesserungen, die schnellere Schreibvorgänge in Parquet-Dateien ermöglichen. Der traditionelle Writer berechnet vor dem

Schreiben ein Schema. Das Parquet-Format speichert das Schema nicht in einer schnell abrufbaren Form, daher kann dies einige Zeit dauern. Mit dem AWS Glue Parquet Writer ist kein vorberechnetes Schema erforderlich. Der Writer berechnet und modifiziert das Schema dynamisch, sobald Daten eingehen.

Beachten Sie die folgenden Einschränkungen, wenn Sie `useGlueParquetWriter` angeben:

- Der Writer unterstützt nur die Schemaentwicklung (z. B. das Hinzufügen oder Entfernen von Spalten), aber nicht das Ändern von Spaltentypen, wie bei `ResolveChoice`.
- Der Writer unterstützt kein leeres Schreiben, z. DataFrames B. um eine reine Schemadatei zu schreiben. Bei der Integration mit dem AWS Glue-Datenkatalog per Einstellung wird der Datenkatalog nicht aktualisiert`enableUpdateCatalog=True`, wenn versucht DataFrame wird, ein leeres Feld zu schreiben. Dies führt dazu, dass im Datenkatalog eine Tabelle ohne Schema erstellt wird.

Wenn für Ihre Transformation diese Einschränkungen nicht erforderlich sind, sollte das Einschalten des AWS Glue Parquet-Writers die Leistung erhöhen.

Verwenden des XML-Formats in AWS Glue

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im XML-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des XML-Formats. Dieses Format stellt hochgradig konfigurierbare, fest definierte Datenstrukturen dar, die nicht zeilen- oder spaltenbasiert sind. XML ist hochgradig standardisiert. Eine Einführung in das Format durch die Standardautorität finden Sie unter XML Essentials.

Sie können AWS Glue verwenden, um XML-Dateien aus Amazon S3 `bzip` sowie `gzip` Archive, die XML-Dateien enthalten, zu lesen. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen AWS Glue-Funktionen die XML-Formatoption unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Nicht unterstützt	Nicht unterstützt	Unterstützt	Unterstützt

Beispiel: Lesen von XML aus S3

Der XML-Reader akzeptiert einen XML-Tag-Namen. Es untersucht Elemente mit diesem Tag in der Eingabe, um daraus ein Schema abzuleiten, und füllt a DynamicFrame mit entsprechenden Werten auf. Die AWS Glue-XML-Funktionalität verhält sich ähnlich wie die XML-Datenquelle für Apache Spark. Möglicherweise können Sie einen Einblick in das grundlegende Verhalten gewinnen, indem Sie diesen Reader mit der Dokumentation dieses Projekts vergleichen.

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den XML-Dateien oder -Ordnern, die Sie lesen möchten, sowie einige Informationen zu Ihrer XML-Datei. Sie benötigen auch das Tag für das XML-Element, das Sie lesen möchten, `xmlTag`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="xml"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben. Sie können weiter konfigurieren, wie der Reader mit S3 in `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:S3-Verbindungsparameter. Verwenden Sie in Ihrem `format_options` den `rowTag`-Schlüssel, um `xmlTag` anzugeben. Sie können weiter konfigurieren, wie der Reader mit XML-Dateien in `format_options` interagiert. Einzelheiten finden Sie in der XML-Konfigurationsreferenz.

Das folgende AWS -Glue-ETL-Skript zeigt den Prozess des Lesens von XML-Dateien oder -Ordnern aus S3.

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
# Example: Read XML from S3
# Set the rowTag option to configure the reader.

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext.getOrCreate()
```

```
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="xml",
    format_options={"rowTag": "xmlTag"},
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame = spark.read\
    .format("xml")\
    .option("rowTag", "xmlTag")\
    .load("s3://s3path")
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
// Example: Read XML from S3
// Set the rowTag option to configure the reader.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkSession

val glueContext = new GlueContext(SparkContext.getOrCreate())
val sparkSession: SparkSession = glueContext.getSparkSession

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"rowTag": "xmlTag"}"""),
      connectionType="s3",
      format="xml",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
}
```

Sie können es auch DataFrames in einem Skript (`org.apache.spark.sql.DataFrame`) verwenden.

```
val dataFrame = spark.read
  .option("rowTag", "xmlTag")
  .format("xml")
  .load("s3://s3path“)
```

XML-Konfigurationsreferenz

Sie können Folgendes verwenden, `format_options` wo immer die AWS Glue-Bibliotheken dies angeben`format="xml"`:

- `rowTag` – Gibt an, dass der XML-Tag in der Datei als Zeile zu behandeln ist. Zeilen-Tags können nicht selbstschließend sein.
  - Typ: Text, Erforderlich
- `encoding` – Gibt die Zeichencodierung an. Dies kann der Name oder Alias eines von unserer Laufzeitumgebung unterstützten Zeichensatzes sein. Wir geben keine spezifischen Garantien für die Unterstützung der Codierung, aber wichtige Kodierungen sollten funktionieren.
  - Typ: Text, Standard: `"UTF-8"`
- `excludeAttribute` – Gibt an, ob Sie Attribute in Elementen ausschließen möchten oder nicht.
  - Typ: Boolesch, Standard: `false`
- `treatEmptyValuesAsNulls` – Gibt an, ob Leerzeichen als Nullwert behandelt werden sollen.
  - Typ: Boolesch, Standard: `false`
- `attributePrefix` – Ein Präfix für Attribute, um sie vom Text der untergeordneten Elemente zu unterscheiden. Das Präfix wird für Feldnamen verwendet.
  - Typ: Text, Standard: `"_"`
- `valueTag` – Das Tag, das für einen Wert verwendet wird, wenn das Element Attribute enthält, die kein untergeordnetes Element haben.
  - Typ: Text, Standard: `"_VALUE"`
- `ignoreSurroundingSpaces` – Gibt an, ob das Leerzeichen, das die Werte umgibt, ignoriert werden soll.
  - Typ: Boolesch, Standard: `false`
- `withSchema` – Enthält das erwartete Schema in Situationen, in denen Sie das abgeleitete Schema überschreiben möchten. Wenn Sie diese Option nicht verwenden, AWS Glue leitet das Schema aus den XML-Daten ab.

- Typ: Text, Standard: Nicht zutreffend
- Der Wert sollte ein JSON-Objekt sein, das ein `StructType` darstellt.

Geben Sie das XML-Schema manuell an

Beispiel eines manuellen XML-Schemas

Dies ist ein Beispiel für die Verwendung der Formatoption `withSchema`, um das Schema für XML-Daten anzugeben.

```
from awsglue.gluetypes import *

schema = StructType([
  Field("id", IntegerType()),
  Field("name", StringType()),
  Field("nested", StructType([
    Field("x", IntegerType()),
    Field("y", StringType()),
    Field("z", ChoiceType([IntegerType(), StringType()]))
  ]))
])

datasource0 = create_dynamic_frame_from_options(
    connection_type,
    connection_options={"paths": ["s3://xml_bucket/someprefix"]},
    format="xml",
    format_options={"withSchema": json.dumps(schema.jsonValue())},
    transformation_ctx = ""
)
```

Verwenden des Avro-Formats in Glue AWS

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im Avro-Datenformat gespeichert oder transportiert werden, werden Ihnen in diesem Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vorgestellt.

AWS Glue unterstützt die Verwendung des Avro-Formats. Dieses Format ist ein leistungsorientiertes, zeilenbasiertes Datenformat. Eine Einführung in das Format durch die Standardautorität finden Sie unter der Apache-Avro 1.8.2 Dokumentation.

Sie können AWS Glue verwenden, um Avro-Dateien von Amazon S3 und aus Streaming-Quellen zu lesen sowie Avro-Dateien auf Amazon S3 zu schreiben. Sie können `bzip2`- und `gzip`-Archive mit Avro-Dateien aus S3 lesen und schreiben. Darüber hinaus können Sie `deflate`-, `snappy`-, und `xz`-Archive schreiben, die Avro-Dateien enthalten. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen AWS Glue-Operationen die Avro-Formatoption unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Unterstützt	Unterstützt*	Nicht unterstützt	Unterstützt

*Mit Einschränkungen unterstützt. Weitere Informationen finden Sie unter the section called “Hinweise und Einschränkungen für Avro-Streaming-Quellen”.

Beispiel: Lesen von Avro-Dateien oder Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den Avro-Dateien oder Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="avro"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben. Sie können konfigurieren, wie der Reader mit S3 in der `connection_options` interagiert. Einzelheiten finden Sie unter Datenformatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue:the section called “S3-Verbindungsparameter”. Sie können konfigurieren, wie der Reader Avro-Dateien in Ihrem `format_options` interpretiert. Einzelheiten finden Sie in der Avro-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Lesens von Avro-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="avro"
)
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="avro",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
```

Beispiel: Schreiben von Avro-Dateien und -Ordnern in S3

Voraussetzungen: Sie benötigen ein initialisiertes DataFrame (`dataFrame`) oder DynamicFrame (`dynamicFrame`). Sie benötigen auch Ihren erwarteten S3-Ausgabepfad, `s3path`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="avro"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um Ihren `s3path` anzugeben. Sie können die Art und Weise, wie der Writer mit S3 in `connection_options` interagiert, weiter verändern. Einzelheiten finden Sie unter Datenformatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue:the section called “S3-Verbindungsparameter”. Sie können ändern, wie der Schreiber Avro-Dateien in Ihren `format_options` interpretiert. Einzelheiten finden Sie in der Avro-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Schreibens von Avro-Dateien oder -Ordnern nach S3.

Python

Verwenden Sie für dieses Beispiel die Methode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="avro",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Scala

Verwenden Sie für dieses Beispiel die getSinkWithFormat-Methode.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="avro"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Avro-Konfigurationsreferenz

Sie können die folgenden `format_options` Werte überall dort verwenden, wo AWS Glue-Bibliotheken dies angeben`format="avro"`:

- `version` – Gibt die zu unterstützende Version des Apache Avro Reader/Writer-Formats an. Der Standardwert ist "1.7". Sie können `format_options={"version": “1.8”}` angeben, um das Lesen und Schreiben des logischen Avro-Typs zu ermöglichen. Weitere Informationen finden Sie unter Apache Avro 1.7.7 Specification und Apache Avro 1.8.2 Specification.

  Der Apache Avro 1.8-Konnektor unterstützt die folgenden Konvertierungen für logische Typen:

Für den Leser: Diese Tabelle zeigt die Konvertierung zwischen dem Avro-Datentyp (logischer Typ und primitiver Avro-Typ) und AWS Glue `DynamicFrame`Datentyp für Avro Reader 1.7 und 1.8.

Avro-Datentyp: Logischer Typ	Avro-Datentyp: Primitiver Avro-Typ	GlueDynamicFrame Datentyp: Avro Reader 1.7	GlueDynamicFrame Datentyp: Avro Reader 1.8
Dezimal	bytes	BINARY	Dezimal
Dezimal	Fixed	BINARY	Dezimal
Datum	int	INT	Datum
Zeit (Millisekunde)	int	INT	INT
Zeit (Mikrosekunde)	long	LONG	LONG
Zeitstempel (Millisekunde)	long	LONG	Zeitstempel
Zeitstempel (Mikrosekunde)	long	LONG	LONG
Dauer (kein logischer Typ)	Fixed von 12	BINARY	BINARY

Für den Autor: Diese Tabelle zeigt die Konvertierung zwischen AWS Glue `DynamicFrame`Datentyp und Avro-Datentyp für Avro Writer 1.7 und 1.8.

AWS Glue **`DynamicFrame`** - Datentyp	Avro-Datentyp: Avro Writer 1.7	Avro-Datentyp: Avro Writer 1.8
Dezimal	String	Dezimalwert
Datum	String	date
Zeitstempel	String	timestamp-micros

Avro Spark-Unterstützung DataFrame

Um Avro über die DataFrame Spark-API verwenden zu können, müssen Sie das Spark Avro-Plugin für die entsprechende Spark-Version installieren. Welche Version von Spark in Ihrem Job verfügbar ist, hängt von Ihrer AWS Glue-Version ab. Weitere Informationen zu den Spark-Versionen erhalten Sie unter the section called “AWS Glue Versionen”. Dieses Plugin wird von Apache verwaltet, wir geben keine spezifischen Garantien für den Support.

Verwenden AWS Sie in Glue 2.0 Version 2.4.3 des Spark Avro-Plugins. Sie finden dieses JAR auf Maven Central, siehe org.apache.spark:spark-avro_2.12:2.4.3.

Verwenden AWS Sie in Glue 3.0 Version 3.1.1 des Spark Avro-Plugins. Sie finden dieses JAR auf Maven Central, siehe org.apache.spark:spark-avro_2.12:3.1.1.

Verwenden Sie den Job-Parameter, um zusätzliche Elemente JARs in einen AWS `--extra-jars` Glue-ETL-Job aufzunehmen. Informationen zu Auftragsparametern finden Sie unter the section called “Auftragsparameter”. Sie können diesen Parameter auch in AWS Management Console konfigurieren.

Verwenden des GrokLog-Formats in Glue AWS

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten in einem locker strukturierten Klartextformat gespeichert oder transportiert werden, werden Ihnen in diesem Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue-through-Grok-Mustern vorgestellt.

AWS Glue unterstützt die Verwendung von Grok-Mustern. Grok-Muster ähneln Erfassungsgruppen mit regulären Ausdrücken. Sie erkennen Muster von Zeichenfolgen in einer Klartextdatei und geben ihnen einen Typ und Zweck. In AWS Glue besteht ihr Hauptzweck darin, Protokolle zu lesen. Eine Einführung in den Grok durch die Autoren finden Sie unter Logstash-Referenz: Grok-Filter-Plugin.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Nicht zutreffend	Unterstützt	Unterstützt	Nicht unterstützt

grokLog-Konfigurationsreferenz

Sie können folgende `format_options`-Werte mit `format="grokLog"` verwenden:

- `logFormat` – Gibt das Grok-Muster an, das mit dem Protokollformat übereinstimmt.
- `customPatterns` – Gibt zusätzliche hier verwendete Grok-Muster an.
- `MISSING` – Gibt das Signal für die Identifizierung fehlender Werte an. Der Standardwert ist `'-'`.
- `LineCount` – Gibt die Anzahl der Zeilen in den einzelnen Protokolldatensätzen an. Der Standard ist `'1'`. Derzeit werden nur einzeilige Datensätze unterstützt.
- `StrictMode` – Ein boolescher Wert, der angibt, ob der Strict-Modus aktiviert ist. Im strikten Modus führt der Reader keine automatische Typkonvertierung oder -wiederherstellung durch. Der Standardwert ist `"false"`.

Das Ion-Format in AWS Glue verwenden

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im Ion-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des Ion-Formats. Dieses Format stellt Datenstrukturen (die nicht zeilen- oder spaltenbasiert sind) in austauschbaren Binär- und Klartext-Darstellungen dar. Eine Einführung in das Format durch die Autoren finden Sie unter Amazon Ion. (Weitere Informationen finden Sie in der Amazon Ion-Spezifikation.)

Sie können AWS Glue verwenden, um Ion-Dateien von Amazon S3 zu lesen. Sie können `bzip`- und `gzip`-Archive mit Ion-Dateien aus S3 lesen. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen AWS Glue-Operationen die Ion-Formatoption unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Nicht unterstützt	Nicht unterstützt	Unterstützt	Nicht unterstützt

Beispiel: Lesen von Ion-Dateien und Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den Ion-Dateien oder -Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="json"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um Ihren `s3path` anzugeben. Sie können konfigurieren, wie der Reader mit S3 in der `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:the section called “S3-Verbindungsparameter”.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Lesens von Ion-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
# Example: Read ION from S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
```

```
    connection_options={"paths": ["s3://s3path"]},
    format="ion"
)
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
// Example: Read ION from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="ion",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Ion-Konfigurationsreferenz

Es gibt keine `format_options`-Werte für `format="ion"`.

Verwenden des JSON-Formats in AWS Glue

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im JSON-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des JSON-Formats. Dieses Format stellt Datenstrukturen mit einheitlicher Form, aber flexiblem Inhalt dar, die nicht zeilen- oder spaltenbasiert sind. JSON wird durch parallel Standards definiert, die von mehreren Behörden herausgegeben werden, von denen

eine ECMA-404 ist. Eine Einführung in das Format durch eine häufig referenzierte Quelle finden Sie unter Einführung in JSON.

Sie können AWS Glue verwenden, um JSON-Dateien aus Amazon S3 `bzip` sowie `gzip` komprimierte JSON-Dateien zu lesen. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen	
Unterstützt	Unterstützt	Unterstützt	Unterstützt	Unterstützt	

Beispiel: Lesen von JSON-Dateien oder Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den JSON-Dateien oder Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="json"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um Ihren `s3path` anzugeben. Sie können in den Verbindungsoptionen weiter ändern, wie Ihr Lesevorgang s3 durchquert, siehe the section called “S3-Verbindungsparameter” für Einzelheiten. Sie können konfigurieren, wie der Reader JSON-Dateien in Ihrem `format_options` interpretiert. Einzelheiten finden Sie in der JSON-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Prozess des Lesens von JSON-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
# Example: Read JSON from S3
# For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
# For show, we also handle a JSON where a single entry spans multiple lines
# Consider whether optimizePerformance is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="json",
    format_options={
        "jsonPath": "$.id",
        "multiline": True,
        # "optimizePerformance": True, -> not compatible with jsonPath, multiline
    }
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame = spark.read\
    .option("multiline", "true")\
    .json("s3://s3path")
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
// Example: Read JSON from S3
// For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
// For show, we also handle a JSON where a single entry spans multiple lines
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"jsonPath": "$.id", "multiline": true,
 "optimizePerformance":false}"""),
```

```
      connectionType="s3",
      format="json",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
val dataFrame = spark.read
    .option("multiline", "true")
    .json("s3://s3path")
```

Beispiel: Schreiben von JSON-Dateien und -Ordnern in S3

Voraussetzungen: Sie benötigen ein initialisiertes DataFrame (`dataFrame`) oder DynamicFrame (`dynamicFrame`). Sie benötigen auch Ihren erwarteten S3-Ausgabepfad, `s3path`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="json"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um `s3path` anzugeben. Sie können die Art und Weise, wie der Writer mit S3 in `connection_options` interagiert, weiter verändern. Einzelheiten finden Sie unter Datenformatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue:the section called “S3-Verbindungsparameter”. Sie können konfigurieren, wie der Writer JSON-Dateien in Ihrem `format_options` interpretiert. Einzelheiten finden Sie in der JSON-Konfigurationsreferenz.

Das folgende AWS Glue-ETL-Skript zeigt den Prozess des Schreibens von JSON-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode write_dynamic_frame.from_options.

```
# Example: Write JSON to S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
```

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="json"
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
df.write.json("s3://s3path/")
```

Scala

Verwenden Sie für dieses Beispiel die getSinkWithFormat-Methode.

```
// Example: Write JSON to S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="json"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Sie können auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
df.write.json("s3://s3path")
```

JSON-Konfigurationsreferenz

Sie können folgende `format_options`-Werte mit `format="json"` verwenden:

- `jsonPath`— Ein JsonPathAusdruck, der ein Objekt identifiziert, das in Datensätze eingelesen werden soll. Dies ist besonders nützlich, wenn eine Datei Datensätze enthält, die in einem äußeren Array verschachtelt sind. Der folgende JsonPath Ausdruck zielt beispielsweise auf das `id` Feld eines JSON-Objekts ab.

```
format="json", format_options={"jsonPath": "$.id"}
```

- `multiline` – Ein boolescher Wert, der angibt, ob ein einzelner Datensatz mehrere Zeilen umfassen kann. Dies kommt vor, wenn ein Feld ein Neue-Zeile-Zeichen in Anführungszeichen enthält. Sie müssen diese Option auf `"true"` setzen, wenn ein Datensatz mehrere Zeilen umfasst. Der Standardwert ist `"false"`. Dies ermöglicht eine rigorosere Dateiaufteilung während der Analyse.
- `optimizePerformance` – Ein boolescher Wert, der angibt, ob der erweiterte SIMD-JSON-Reader zusammen mit Apache Arrow basierten spaltenförmigen Speicherformaten verwendet werden soll. Nur verfügbar in AWS Glue 3.0. Nicht kompatibel mit `multiline` oder `jsonPath`. Wenn Sie eine dieser Optionen angeben, wird AWS Glue angewiesen, auf den Standardleser zurückzugreifen.
- `withSchema` – Ein String-Wert, der ein Tabellenschema in dem in the section called “Angeben von XML-Schema” beschriebenen Format festlegt. Wird nur mit `optimizePerformance` beim Lesen von Nicht-Catalog-Verbindungen verwendet.

Verwendung vektorisierter SIMD-JSON-Reader mit Apache-Arrow-Spaltenformat

AWS Glue Version 3.0 fügt einen vektorisierten Reader für JSON-Daten hinzu. Er arbeitet unter bestimmten Bedingungen doppelt so schnell als der Standard-Reader. Dieser Reader weist bestimmte Einschränkungen auf, die Benutzer vor der Verwendung beachten sollten und die in diesem Abschnitt dokumentiert werden.

Um den optimierten Reader zu verwenden, setzen Sie `"optimizePerformance"` auf True in der `format_options`- oder Tabelleneigenschaft. Sie müssen außerdem `withSchema` angeben, sofern nicht aus dem Katalog gelesen wird. `withSchema` erwartet eine Eingabe wie in der the section called “Angeben von XML-Schema” beschrieben

```
// Read from S3 data source
glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path"]},
    format = "json",
```

```
    format_options={
        "optimizePerformance": True,
        "withSchema": SchemaString
        })

// Read from catalog table
glueContext.create_dynamic_frame.from_catalog(
    database = database,
    table_name = table,
    additional_options = {
    // The vectorized reader for JSON can read your schema from a catalog table
 property.
        "optimizePerformance": True,
        })
```

Weitere Informationen zum Gebäude a *SchemaString* in der AWS Glue-Bibliothek finden Sie unterthe section called “Typen”.

Einschränkungen für den vektorisierten CSV-Reader

Es gelten die folgenden Einschränkungen:

- JSON-Elemente mit verschachtelten Objekten oder Array-Werten werden nicht unterstützt. Falls angegeben, AWS greift Glue auf den Standardleser zurück.
- Ein Schema muss bereitgestellt werden, entweder aus dem Katalog oder mit `withSchema`.
- Nicht kompatibel mit `multiline` oder `jsonPath`. Wenn Sie eine dieser Optionen angeben, wird AWS Glue angewiesen, auf den Standardleser zurückzugreifen.
- Das Bereitstellen von Eingabedatensätzen, die nicht mit dem Eingabeschema übereinstimmen, führt zum Fehlschlagen des Readers.
- Fehler-Aufzeichnungen werden nicht erstellt.
- Das Lesen von JSON-Dateien mit MultiByte-Zeichen (wie japanische oder chinesische Zeichen) wird nicht unterstützt.

Verwenden des ORC-Formats in Glue AWS

AWS Glue ruft Daten aus Quellen ab und schreibt Daten auf Ziele, die in verschiedenen Datenformaten gespeichert und transportiert werden. Wenn Ihre Daten im ORC-Datenformat gespeichert oder transportiert werden, stellt Ihnen dieses Dokument die verfügbaren Funktionen für die Verwendung Ihrer Daten in AWS Glue vor.

AWS Glue unterstützt die Verwendung des ORC-Formats. Dieses Format ist ein leistungsorientiertes, spaltenbasiertes Datenformat. Eine Einführung in das Format durch die Standardautorität finden Sie unter Apache Orc.

Sie können AWS Glue verwenden, um ORC-Dateien von Amazon S3 und aus Streaming-Quellen zu lesen sowie ORC-Dateien auf Amazon S3 zu schreiben. Sie können `bzip`- und `gzip`-Archive mit ORC-Dateien aus S3 lesen und schreiben. Sie konfigurieren das Komprimierungsverhalten auf S3-Verbindungsparameter statt in der auf dieser Seite besprochenen Konfiguration.

Die folgende Tabelle zeigt, welche gängigen AWS Glue-Operationen die ORC-Formatoption unterstützen.

Lesen	Write (Schreiben)	Streaming gelesen	Gruppieren von kleinen Dateien	Auftrags-Lesezeichen
Unterstützt	Unterstützt	Unterstützt	Nicht unterstützt	Unterstützt*

* Wird in AWS Glue Version 1.0+ unterstützt

Beispiel: Lesen von ORC-Dateien oder Ordnern aus S3

Voraussetzungen: Sie benötigen die S3-Pfade (`s3path`) zu den ORC-Dateien oder -Ordnern, die Sie lesen möchten.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="orc"` an. Verwenden Sie in Ihrem `connection_options` den `paths`-Schlüssel, um Ihren `s3path` anzugeben. Sie können konfigurieren, wie der Reader mit S3 in der `connection_options` interagiert. Einzelheiten finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue:the section called “S3-Verbindungsparameter”.

Das folgende AWS Glue-ETL-Skript zeigt den Vorgang des Lesens von ORC-Dateien oder -Ordnern aus S3:

Python

Verwenden Sie für dieses Beispiel die Methode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="orc"
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
dataFrame = spark.read\
    .orc("s3://s3path")
```

Scala

Verwenden Sie für dieses Beispiel die Operation getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="orc",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
val dataFrame = spark.read
    .orc("s3://s3path")
```

Beispiel: Schreiben von ORC-Dateien und -Ordnern in S3

Voraussetzungen: Sie benötigen ein initialisiertes DataFrame (`dataFrame`) oder DynamicFrame (`dynamicFrame`). Sie benötigen auch Ihren erwarteten S3-Ausgabepfad, `s3path`.

Konfiguration: Geben Sie in Ihren Funktionsoptionen `format="orc"` an. Verwenden Sie in Ihren Verbindungsoptionen die `paths`-Schlüssel zum Angeben von `s3path`. Sie können die Art und Weise, wie der Writer mit S3 in `connection_options` interagiert, weiter verändern. Einzelheiten finden Sie unter Datenformatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue:the section called “S3-Verbindungsparameter”. Das folgende Code-Beispiel veranschaulicht den Prozess:

Python

Verwenden Sie für dieses Beispiel die Methode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="orc",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
df.write.orc("s3://s3path/")
```

Scala

Verwenden Sie für dieses Beispiel die getSinkWithFormat-Methode.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext
```

```
object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="orc"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Sie können es auch DataFrames in einem Skript (`pyspark.sql.DataFrame`) verwenden.

```
df.write.orc("s3://s3path/")
```

ORC-Konfigurationsreferenz

Es gibt keine `format_options`-Werte für `format="orc"`. Es können jedoch alle Optionen, die vom zugrunde liegenden SparkSQL-Code akzeptiert werden, im Rahmen des `connection_options`-Map-Parameters übergeben werden.

Verwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs

Open-Source-Data-Lake-Frameworks vereinfachen die inkrementelle Datenverarbeitung für Dateien, die Sie in Data Lakes speichern, die auf Amazon S3 basieren. AWS Glue 3.0 und höher unterstützt die folgenden Open-Source-Data-Lake-Frameworks:

- Apache Hudi
- Linux Foundation Delta Lake
- Apache Iceberg

Wir bieten native Unterstützung für diese Frameworks an, sodass Sie Daten, die Sie in Amazon S3 speichern, transaktionskonsistent lesen und schreiben können. Es ist nicht erforderlich, einen separaten Konnektor zu installieren oder zusätzliche Konfigurationsschritte durchzuführen, um diese Frameworks in AWS Glue -ETL-Aufträgen zu verwenden.

Wenn Sie Datensätze über die verwalten AWS Glue Data Catalog, können Sie AWS Glue Methoden zum Lesen und Schreiben von Data-Lake-Tabellen mit Spark verwenden. DataFrames Sie können Amazon S3 S3-Daten auch mithilfe der DataFrame Spark-API lesen und schreiben.

In diesem Video erfahren Sie mehr über die Grundlagen der Funktionsweise von Apache Hudi, Apache Iceberg und Delta Lake. Sie erfahren, wie Sie Daten in Ihren Data Lake einfügen, aktualisieren und löschen und wie jedes dieser Frameworks funktioniert.

Themen

- Einschränkungen
- Verwenden des Hudi-Frameworks in AWS Glue
- Verwenden des Delta Lake-Frameworks in AWS Glue
- Verwenden des Iceberg-Frameworks in AWS Glue

Einschränkungen

Beachten Sie die folgenden Einschränkungen, bevor Sie Data Lake-Frameworks mit verwenden. AWS Glue

- Die folgenden AWS Glue `GlueContext` Methoden für unterstützen das Lesen und Schreiben von Data Lake-Framework-Tabellen DynamicFrame nicht. Verwenden Sie stattdessen die `GlueContext` Methoden für DataFrame oder die DataFrame Spark-API.
  - `create_dynamic_frame.from_catalog`
  - `write_dynamic_frame.from_catalog`
  - `getDynamicFrame`
  - `writeDynamicFrame`
- Die folgenden `GlueContext` Methoden für DataFrame werden von Lake Formation Permission Control unterstützt:
  - `create_data_frame.from_catalog`
  - `write_data_frame.from_catalog`
  - `getDataFrame`
  - `writeDataFrame`
- Das Gruppieren kleiner Dateien wird nicht unterstützt.
- Auftragslesezeichen werden nicht unterstützt.

- Apache Hudi 0.10.1 für AWS Glue 3.0 unterstützt keine Hudi Merge on Read (MoR) -Tabellen.
- `ALTER TABLE … RENAME TO`ist für Apache Iceberg 0.13.1 für 3.0 nicht verfügbar. AWS Glue

Einschränkungen für Tabellen im Data-Lake-Format, die mit Lake-Formation-Berechtigungen verwaltet werden

Die Data-Lake-Formate sind über Lake Formation Formation-Berechtigungen in AWS Glue ETL integriert. Das Erstellen einer DynamicFrame Verwendung `create_dynamic_frame` wird nicht unterstützt. Weitere Informationen finden Sie in den folgenden Beispielen:

- Beispiel: Lesen und Schreiben einer Iceberg-Tabelle mit Lake-Formation-Berechtigungskontrolle
- Beispiel: Lesen und Schreiben einer Hudi-Tabelle mit Lake-Formation-Berechtigungskontrolle
- Beispiel: Lesen und Schreiben einer Delta-Lake-Tabelle mit Lake-Formation-Berechtigungskontrolle

> **Note**
>
> Die Integration mit AWS Glue ETL über Lake Formation Formation-Berechtigungen für Apache Hudi, Apache Iceberg und Delta Lake wird nur in AWS Glue Version 4.0 unterstützt.

Apache Iceberg bietet die beste Integration mit AWS Glue ETL über Lake Formation Formation-Berechtigungen. Es unterstützt fast alle Operationen und beinhaltet SQL-Unterstützung.

Hudi unterstützt die meisten grundlegenden Operationen mit Ausnahme von Verwaltungsoperationen. Dies liegt daran, dass diese Optionen im Allgemeinen durch das Schreiben von DataFrames erfolgen und über `additional_options` spezifiziert werden. Sie müssen AWS Glue APIs to create DataFrames für Ihre Operationen verwenden, da SparkSQL nicht unterstützt wird.

Delta Lake unterstützt nur das Lesen, Anhängen und Überschreiben von Tabellendaten. Delta Lake erfordert die Verwendung eigener Bibliotheken, um verschiedene Aufgaben wie Aktualisierungen ausführen zu können.

Die folgenden Funktionen sind für Iceberg-Tabellen, die mit Lake-Formation-Berechtigungen verwaltet werden, nicht verfügbar.

- Verdichtung mit ETL AWS Glue
- Spark SQL-Unterstützung über AWS Glue ETL

Im Folgenden sind die Einschränkungen von Hudi-Tabellen aufgeführt, die mit Lake-Formation-Berechtigungen verwaltet werden:

- Entfernen verwaister Dateien

Im Folgenden sind die Einschränkungen von Delta-Lake-Tabellen aufgeführt, die mit Lake-Formation-Berechtigungen verwaltet werden:

- Alle Features außer dem Einfügen und Lesen von Delta-Lake-Tabellen.

Verwenden des Hudi-Frameworks in AWS Glue

AWS Glue 3.0 und höher unterstützt das Apache Hudi-Framework für Data Lakes. Hudi ist ein Open-Source-Framework für Data-Lake-Speicher, das die inkrementelle Datenverarbeitung und die Entwicklung von Datenpipelines vereinfacht. Dieses Thema behandelt verfügbare Funktionen für die Verwendung Ihrer Daten in AWS Glue, wenn Sie Ihre Daten in einer Hudi-Tabelle transportieren oder speichern. Weitere Informationen zu Hudi finden Sie in der offiziellen Apache-Hudi-Dokumentation.

Sie können AWS Glue verwenden, um Lese- und Schreibvorgänge an Hudi-Tabellen in Amazon S3 durchzuführen, oder mithilfe des AWS Glue-Datenkatalogs mit Hudi-Tabellen arbeiten. Zusätzliche Vorgänge wie Einfügen, Aktualisieren und alle Apache-Spark-Vorgänge werden ebenfalls unterstützt.

> **Note**
>
> Die Apache Hudi 0.15.0-Implementierung in AWS Glue 5.0 macht HUDI-7001 intern rückgängig. Es zeigt nicht die Regression, die sich auf die Generierung komplexer Schlüssel bezieht, wenn der Datensatzschlüssel aus einem einzigen Feld besteht. Dieses Verhalten unterscheidet sich jedoch von OSS Apache Hudi 0.15.0.
> Apache Hudi 0.10.1 für AWS Glue 3.0 unterstützt keine Hudi Merge on Read (MoR) - Tabellen.

In der folgenden Tabelle ist die Hudi-Version aufgeführt, die in jeder AWS Glue-Version enthalten ist.

AWS Ausführung mit Glue	Unterstützte Hudi-Versionen
5.0	0.15.0

AWS Ausführung mit Glue	Unterstützte Hudi-Versionen
4,0	0.12.1
3.0	0.10.1

Weitere Informationen zu den von AWS Glue unterstützten Data Lake-Frameworks finden Sie unterVerwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

Aktivieren von Hudi

Führen Sie die folgenden Aufgaben aus, um Hudi for AWS Glue zu aktivieren:

- Geben Sie `hudi` als Wert für den Auftragsparameter `--datalake-formats` an. Weitere Informationen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.
- Erstellen Sie einen Schlüssel, der `--conf` nach Ihrem AWS Glue-Job benannt ist, und setzen Sie ihn auf den folgenden Wert. Alternativ können Sie die folgende Konfiguration mit `SparkConf` in Ihrem Skript festlegen. Diese Einstellungen helfen Apache Spark bei der korrekten Handhabung von Hudi-Tabellen.

```
spark.serializer=org.apache.spark.serializer.KryoSerializer
```

- Die Unterstützung Lake Formation Formation-Berechtigungen für Hudi ist standardmäßig für AWS Glue 4.0 aktiviert. Für das Lesen/Schreiben in Hudi-Tabellen, die bei Lake Formation registriert sind, ist keine zusätzliche Konfiguration erforderlich. Um eine registrierte Hudi-Tabelle lesen zu können, muss die IAM-Rolle des AWS Glue-Jobs über die SELECT-Berechtigung verfügen. Um in eine registrierte Hudi-Tabelle zu schreiben, muss die IAM-Rolle des AWS Glue-Jobs über die SUPER-Berechtigung verfügen. Weitere Informationen zur Verwaltung von Lake-Formation-Berechtigungen finden Sie unter Granting and revoking permissions on Data Catalog resources.

Verwenden einer anderen Hudi-Version

Um eine Version von Hudi zu verwenden, die AWS Glue nicht unterstützt, geben Sie Ihre eigenen Hudi-JAR-Dateien mit dem `--extra-jars` Job-Parameter an. Schließen Sie `hudi` nicht als Wert für den Auftragsparameter `--datalake-formats` ein. Wenn Sie AWS Glue 5.0 verwenden, müssen Sie den `--user-jars-first true` Job-Parameter festlegen.

Beispiel: Schreiben Sie eine Hudi-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog

Dieses Beispielskript zeigt, wie eine Hudi-Tabelle in Amazon S3 geschrieben und die Tabelle im AWS Glue-Datenkatalog registriert wird. Das Beispiel verwendet das Hive-Sync-Tool von Hudi zum Registrieren der Tabelle.

> **Note**
>
> In diesem Beispiel müssen Sie den `--enable-glue-datacatalog` Job-Parameter festlegen, um den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore verwenden zu können. Weitere Informationen hierzu finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Python

```
# Example: Create a Hudi table from a DataFrame
# and register the table to Glue Data Catalog

additional_options={
    "hoodie.table.name": "<your_table_name>",
    "hoodie.database.name": "<your_database_name>",
    "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
    "hoodie.datasource.write.operation": "upsert",
    "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
    "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
    "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
    "hoodie.datasource.write.hive_style_partitioning": "true",
    "hoodie.datasource.hive_sync.enable": "true",
    "hoodie.datasource.hive_sync.database": "<your_database_name>",
    "hoodie.datasource.hive_sync.table": "<your_table_name>",
    "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
    "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
    "hoodie.datasource.hive_sync.use_jdbc": "false",
    "hoodie.datasource.hive_sync.mode": "hms",
    "path": "s3://<s3Path/>"
}

dataFrame.write.format("hudi") \
    .options(**additional_options) \
```

```
    .mode("overwrite") \
    .save()
```

Scala

```
// Example: Example: Create a Hudi table from a DataFrame
// and register the table to Glue Data Catalog

val additionalOptions = Map(
  "hoodie.table.name" -> "<your_table_name>",
  "hoodie.database.name" -> "<your_database_name>",
  "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
  "hoodie.datasource.write.operation" -> "upsert",
  "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
  "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
  "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
  "hoodie.datasource.write.hive_style_partitioning" -> "true",
  "hoodie.datasource.hive_sync.enable" -> "true",
  "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
  "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
  "hoodie.datasource.hive_sync.partition_fields" -> "<your_partitionkey_field>",
  "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
  "hoodie.datasource.hive_sync.use_jdbc" -> "false",
  "hoodie.datasource.hive_sync.mode" -> "hms",
  "path" -> "s3://<s3Path/>")

dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("append")
  .save()
```

Beispiel: Lesen einer Hudi-Tabelle aus Amazon S3 mithilfe des AWS Glue-Datenkatalogs

In diesem Beispiel wird die Hudi-Tabelle gelesen, die Sie in Beispiel: Schreiben Sie eine Hudi-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog aus Amazon S3 erstellt haben.

> **Note**
>
> In diesem Beispiel müssen Sie den `--enable-glue-datacatalog` Job-Parameter festlegen, um den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore verwenden zu

können. Weitere Informationen hierzu finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Python

Verwenden Sie für dieses Beispiel die GlueContext.create_data_frame.from_catalog()-Methode.

```
# Example: Read a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

dataFrame = glueContext.create_data_frame.from_catalog(
    database = "<your_database_name>",
    table_name = "<your_table_name>"
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSource-Methode.

```
// Example: Read a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dataFrame = glueContext.getCatalogSource(
      database = "<your_database_name>",
      tableName = "<your_table_name>"
    ).getDataFrame()
  }
}
```

Beispiel: Aktualisieren und Einfügen eines **`DataFrame`** in eine Hudi-Tabelle in Amazon S3

In diesem Beispiel wird der AWS Glue-Datenkatalog verwendet, um eine DataFrame in die Hudi-Tabelle einzufügen, in Beispiel: Schreiben Sie eine Hudi-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog der Sie erstellt haben.

> **Note**
>
> In diesem Beispiel müssen Sie den `--enable-glue-datacatalog` Job-Parameter festlegen, um den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore verwenden zu können. Weitere Informationen hierzu finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Python

Verwenden Sie für dieses Beispiel die `GlueContext.write_data_frame.from_catalog()`-Methode.

```
# Example: Upsert a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame = dataFrame,
    database = "<your_database_name>",
    table_name = "<your_table_name>",
    additional_options={
        "hoodie.table.name": "<your_table_name>",
        "hoodie.database.name": "<your_database_name>",
        "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
        "hoodie.datasource.write.operation": "upsert",
        "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning": "true",
        "hoodie.datasource.hive_sync.enable": "true",
        "hoodie.datasource.hive_sync.database": "<your_database_name>",
```

```
        "hoodie.datasource.hive_sync.table": "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc": "false",
        "hoodie.datasource.hive_sync.mode": "hms"
    }
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSink-Methode.

```
// Example: Upsert a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.JsonOptions
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.database.name" -> "<your_database_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "upsert",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
```

```
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Beispiel: Lesen einer Hudi-Tabelle aus Amazon S3 mit Spark

In diesem Beispiel wird mithilfe der DataFrame Spark-API eine Hudi-Tabelle aus Amazon S3 gelesen.

Python

```
# Example: Read a Hudi table from S3 using a Spark DataFrame

dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Hudi table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Beispiel: Schreiben einer Hudi-Tabelle in Amazon S3 mit Spark

In diesem Beispiel wird eine Hudi-Tabelle mit Spark in Amazon S3 geschrieben.

Python

```
# Example: Write a Hudi table to S3 using a Spark DataFrame

dataFrame.write.format("hudi") \
    .options(**additional_options) \
    .mode("overwrite") \
    .save("s3://<s3Path/>")
```

Scala

```
// Example: Write a Hudi table to S3 using a Spark DataFrame

dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("overwrite")
```

```
    .save("s3://<s3path/>")
```

Beispiel: Lesen und Schreiben einer Hudi-Tabelle mit Lake-Formation-Berechtigungskontrolle

In diesem Beispiel wird mit Lake-Formation-Berechtigungen in einer Hudi-Tabelle gelesen und geschrieben.

1. Erstellen einer Hudi-Tabelle und Registrieren in Lake Formation.
   a. Um die Lake-Formation-Berechtigungskontrolle zu aktivieren, müssen Sie zunächst den Amazon-S3-Tabellenpfad auf Lake Formation registrieren. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Speicherorts. Sie können es entweder über die Lake Formation Formation-Konsole oder über die AWS CLI registrieren:

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
   folder> --use-service-linked-role --region <REGION>
   ```

   Sobald Sie einen Amazon S3 S3-Standort registriert haben, gibt jede AWS Glue-Tabelle, die auf den Standort (oder einen seiner untergeordneten Standorte) zeigt, den Wert für den `IsRegisteredWithLakeFormation` Parameter im `GetTable` Aufruf als wahr zurück.

   b. Erstellen einer Hudi-Tabelle, die über die Spark-DataFrame-API auf den registrierten Amazon-S3-Pfad verweist:

   ```
   hudi_options = {
       'hoodie.table.name': table_name,
       'hoodie.database.name': database_name,
       'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
       'hoodie.datasource.write.recordkey.field': 'product_id',
       'hoodie.datasource.write.table.name': table_name,
       'hoodie.datasource.write.operation': 'upsert',
       'hoodie.datasource.write.precombine.field': 'updated_at',
       'hoodie.datasource.write.hive_style_partitioning': 'true',
       'hoodie.upsert.shuffle.parallelism': 2,
       'hoodie.insert.shuffle.parallelism': 2,
       'path': <S3_TABLE_LOCATION>,
       'hoodie.datasource.hive_sync.enable': 'true',
       'hoodie.datasource.hive_sync.database': database_name,
       'hoodie.datasource.hive_sync.table': table_name,
       'hoodie.datasource.hive_sync.use_jdbc': 'false',
       'hoodie.datasource.hive_sync.mode': 'hms'
   ```

```
}

df_products.write.format("hudi")  \
    .options(**hudi_options)  \
    .mode("overwrite")  \
    .save()
```

2. Erteilen Sie Lake Formation die Erlaubnis für die IAM-Rolle AWS Glue Job. Sie können Berechtigungen entweder über die Lake Formation Formation-Konsole oder über die AWS CLI gewähren. Weitere Informationen finden Sie unter Granting table permissions using the Lake Formation console and the named resource method.
3. Lesen der in Lake Formation registrierten Hudi-Tabelle. Der Code entspricht dem Lesen einer nicht registrierten Hudi-Tabelle. Beachten Sie, dass die IAM-Rolle des AWS Glue-Jobs über die SELECT-Berechtigung verfügen muss, damit der Lesevorgang erfolgreich ist.

```
val dataFrame = glueContext.getCatalogSource(
     database = "<your_database_name>",
     tableName = "<your_table_name>"
   ).getDataFrame()
```

4. Schreiben in eine in Lake Formation registrierte Hudi-Tabelle. Der Code entspricht dem Schreiben in eine nicht registrierte Hudi-Tabelle. Beachten Sie, dass die IAM-Rolle des AWS Glue-Jobs über die SUPER-Berechtigung verfügen muss, damit der Schreibvorgang erfolgreich ist.

```
glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.database.name" -> "<your_database_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "<write_operation>",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
```

```
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
      .writeDataFrame(dataFrame, glueContext)
```

Verwenden des Delta Lake-Frameworks in AWS Glue

AWS Glue 3.0 und höher unterstützt das Delta Lake-Framework der Linux Foundation. Delta Lake ist ein Open-Source-Framework für Data-Lake-Speicher, mit dem Sie ACID-Transaktionen durchführen, die Verarbeitung von Metadaten skalieren und Streaming- und Batch-Datenverarbeitung vereinheitlichen können. In diesem Thema werden verfügbare Funktionen für die Verwendung Ihrer Daten in AWS Glue beschrieben, wenn Sie Ihre Daten in einer Delta Lake-Tabelle transportieren oder speichern. Weitere Informationen zu Delta Lake finden Sie in der offiziellen Delta-Lake-Dokumentation.

Sie können AWS Glue verwenden, um Lese- und Schreibvorgänge an Delta Lake-Tabellen in Amazon S3 durchzuführen, oder mit Delta Lake-Tabellen mithilfe des AWS Glue-Datenkatalogs arbeiten. Zusätzliche Operationen wie Einfügen, Aktualisieren und Lesen und Schreiben von Tabellenbatches werden ebenfalls unterstützt. Wenn Sie Delta-Lake-Tabellen verwenden, haben Sie auch die Möglichkeit, Methoden aus der Delta-Lake-Python-Bibliothek zu verwenden, wie z. B. `DeltaTable.forPath`. Weitere Informationen zur Delta Lake Python-Bibliothek finden Sie in der Python-Dokumentation von Delta Lake.

In der folgenden Tabelle sind die Versionen von Delta Lake aufgeführt, die in jeder AWS Glue-Version enthalten sind.

AWS Ausführung mit Glue	Unterstützte Delta-Lake-Version
5.0	3.3.0
4,0	2.1.0
3.0	1.0.0

Weitere Informationen zu den von AWS Glue unterstützten Data Lake-Frameworks finden Sie unterVerwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

Delta Lake für AWS Glue aktivieren

Führen Sie die folgenden Aufgaben aus, um Delta Lake für AWS Glue zu aktivieren:

- Geben Sie `delta` als Wert für den Auftragsparameter `--datalake-formats` an. Weitere Informationen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.
- Erstellen Sie einen Schlüssel, der `--conf` nach Ihrem AWS Glue-Job benannt ist, und setzen Sie ihn auf den folgenden Wert. Alternativ können Sie die folgende Konfiguration mit `SparkConf` in Ihrem Skript festlegen. Diese Einstellungen helfen Apache Spark bei der korrekten Handhabung von Delta-Lake-Tabellen.

```
spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension --conf
 spark.sql.catalog.spark_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog --
conf
 spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDriverLogStore
```

- Die Unterstützung Lake Formation Formation-Berechtigungen für Delta-Tabellen ist standardmäßig für AWS Glue 4.0 aktiviert. Für das Lesen/Schreiben in Delta-Tabellen, die bei Lake Formation registriert sind, ist keine zusätzliche Konfiguration erforderlich. Um eine registrierte Delta-Tabelle lesen zu können, muss die IAM-Rolle des AWS Glue-Jobs über die SELECT-Berechtigung verfügen. Um in eine registrierte Delta-Tabelle schreiben zu können, muss die IAM-Rolle des AWS Glue-Jobs über die SUPER-Berechtigung verfügen. Weitere Informationen zur Verwaltung von Lake-Formation-Berechtigungen finden Sie unter Granting and revoking permissions on Data Catalog resources.

Verwenden einer anderen Delta-Lake-Version

Um eine Version von Delta Lake zu verwenden, die AWS Glue nicht unterstützt, geben Sie Ihre eigenen Delta Lake-JAR-Dateien mit dem `--extra-jars` Job-Parameter an. Schließen Sie `delta` nicht als Wert für den Auftragsparameter `--datalake-formats` ein. Wenn Sie AWS Glue 5.0 verwenden, müssen Sie den `--user-jars-first true` Job-Parameter festlegen. Um die Delta-Lake-Python-Bibliothek in diesem Fall zu verwenden, müssen Sie die JAR-Dateien der Bibliothek mithilfe des `--extra-py-files`-Auftragsparameters angeben. Die Python-Bibliothek ist in den JAR-Dateien von Delta Lake enthalten.

Beispiel: Schreiben Sie eine Delta Lake-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue Data Catalog

Das folgende AWS Glue-ETL-Skript veranschaulicht, wie Sie eine Delta Lake-Tabelle in Amazon S3 schreiben und die Tabelle im AWS Glue-Datenkatalog registrieren.

Python

```
# Example: Create a Delta Lake table from a DataFrame
# and register the table to Glue Data Catalog

additional_options = {
    "path": "s3://<s3Path>"
}
dataFrame.write \
    .format("delta") \
    .options(**additional_options) \
    .mode("append") \
    .partitionBy("<your_partitionkey_field>") \
    .saveAsTable("<your_database_name>.<your_table_name>")
```

Scala

```
// Example: Example: Create a Delta Lake table from a DataFrame
// and register the table to Glue Data Catalog

val additional_options = Map(
  "path" -> "s3://<s3Path>"
)
dataFrame.write.format("delta")
  .options(additional_options)
  .mode("append")
  .partitionBy("<your_partitionkey_field>")
  .saveAsTable("<your_database_name>.<your_table_name>")
```

Beispiel: Lesen einer Delta Lake-Tabelle aus Amazon S3 mithilfe des AWS Glue-Datenkatalogs

Das folgende AWS Glue-ETL-Skript liest die Delta Lake-Tabelle, in der Sie erstellt habenBeispiel: Schreiben Sie eine Delta Lake-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue Data Catalog.

Python

Verwenden Sie für dieses Beispiel die create_data_frame.from_catalog-Methode.

```
# Example: Read a Delta Lake table from Glue Data Catalog
```

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSource-Methode.

```
// Example: Read a Delta Lake table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Beispiel: Einfügen eines **`DataFrame`** in eine Delta-Lake-Tabelle in Amazon S3 unter Verwendung des AWS Glue Data Catalog

In diesem Beispiel werden Daten in die Delta-Lake-Tabelle eingefügt, die Sie in Beispiel: Schreiben Sie eine Delta Lake-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue Data Catalog erstellt haben.

> **Note**
>
> In diesem Beispiel müssen Sie den `--enable-glue-datacatalog` Job-Parameter festlegen, um den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore verwenden zu können. Weitere Informationen hierzu finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Python

Verwenden Sie für dieses Beispiel die create_data_frame.from_catalog-Methode.

```
# Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSink-Methode.

```
// Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
```

```
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Beispiel: Lesen einer Delta-Lake-Tabelle aus Amazon S3 mit der Spark-API

In diesem Beispiel wird mit der Spark-API eine Delta-Lake-Tabelle aus Amazon S3 gelesen.

Python

```
# Example: Read a Delta Lake table from S3 using a Spark DataFrame

dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Delta Lake table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Beispiel: Schreiben einer Delta-Lake-Tabelle in Amazon S3 mit Spark

In diesem Beispiel wird eine Delta-Lake-Tabelle mit Spark in Amazon S3 geschrieben.

Python

```
# Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta") \
    .options(**additional_options) \
    .mode("overwrite") \
    .partitionBy("<your_partitionkey_field>") \
    .save("s3://<s3Path>")
```

Scala

```
// Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta")
```

```
    .options(additionalOptions)
    .mode("overwrite")
    .partitionBy("<your_partitionkey_field>")
    .save("s3://<s3path/>")
```

Beispiel: Lesen und Schreiben einer Delta-Lake-Tabelle mit Lake-Formation-Berechtigungskontrolle

In diesem Beispiel wird mit Lake-Formation-Berechtigungen in einer Delta-Lake-Tabelle gelesen und geschrieben.

1. Erstellen einer Delta-Tabelle und Registrieren in Lake Formation
   a. Um die Lake-Formation-Berechtigungskontrolle zu aktivieren, müssen Sie zunächst den Amazon-S3-Tabellenpfad auf Lake Formation registrieren. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Speicherorts. Sie können es entweder über die Lake Formation Formation-Konsole oder über die AWS CLI registrieren:

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
   folder> --use-service-linked-role --region <REGION>
   ```

   Sobald Sie einen Amazon S3 S3-Standort registriert haben, gibt jede AWS Glue-Tabelle, die auf den Standort (oder einen seiner untergeordneten Standorte) zeigt, den Wert für den `IsRegisteredWithLakeFormation` Parameter im `GetTable` Aufruf als wahr zurück.

   b. Erstellen einer Delta-Tabelle, die über Spark auf den registrierten Amazon-S3-Pfad verweist:

   > **Note**
   >
   > Im Folgenden sind Python-Beispiele aufgeführt.

   ```
   dataFrame.write \
    .format("delta") \
    .mode("overwrite") \
    .partitionBy("<your_partitionkey_field>") \
    .save("s3://<the_s3_path>")
   ```

   Nachdem die Daten in Amazon S3 geschrieben wurden, verwenden Sie den AWS Glue-Crawler, um eine neue Delta-Katalogtabelle zu erstellen. Weitere Informationen finden Sie unter Einführung der nativen Delta Lake-Tabellenunterstützung mit AWS Glue-Crawlern.

Sie können die Tabelle auch manuell über die AWS `CreateTable` Glue-API erstellen.

2. Erteilen Sie Lake Formation die Erlaubnis für die IAM-Rolle des Jobs AWS Glue. Sie können Berechtigungen entweder über die Lake Formation Formation-Konsole oder über die AWS CLI gewähren. Weitere Informationen finden Sie unter Granting table permissions using the Lake Formation console and the named resource method.
3. Lesen der in Lake Formation registrierten Delta-Tabelle. Der Code entspricht dem Lesen einer nicht registrierten Delta-Tabelle. Beachten Sie, dass die IAM-Rolle des AWS Glue-Jobs über die SELECT-Berechtigung verfügen muss, damit der Lesevorgang erfolgreich ist.

```
# Example: Read a Delta Lake table from Glue Data Catalog

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Schreiben in eine in Lake Formation registrierte Delta-Tabelle. Der Code entspricht dem Schreiben in eine nicht registrierte Delta-Tabelle. Beachten Sie, dass die IAM-Rolle des AWS Glue-Jobs über die SUPER-Berechtigung verfügen muss, damit der Schreibvorgang erfolgreich ist.

   Standardmäßig verwendet AWS Glue den `Append` SaveMode. Sie können das ändern, indem Sie die saveMode-Option in `additional_options` einstellen. Informationen zur saveMode-Unterstützung in Delta-Tabellen finden Sie unter Write to a table.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Verwenden des Iceberg-Frameworks in AWS Glue

AWS Glue 3.0 und höher unterstützt das Apache Iceberg-Framework für Data Lakes. Iceberg bietet ein leistungsstarkes Tabellenformat, das genauso wie eine SQL-Tabelle funktioniert. In diesem Thema werden verfügbare Funktionen für die Verwendung Ihrer Daten in AWS Glue beschrieben,

wenn Sie Ihre Daten in einer Iceberg-Tabelle transportieren oder speichern. Weitere Informationen zu Iceberg finden Sie in der offiziellen Apache-Iceberg-Dokumentation.

Sie können AWS Glue verwenden, um Lese- und Schreibvorgänge an Iceberg-Tabellen in Amazon S3 durchzuführen, oder mithilfe des AWS Glue-Datenkatalogs mit Iceberg-Tabellen arbeiten. Zusätzliche Operationen wie Einfügen und alle Spark-Abfragen, die Spark Writes enthalten, werden ebenfalls unterstützt. Update wird für Iceberg-Tabellen nicht unterstützt.

> **Note**
>
> `ALTER TABLE … RENAME TO` ist für Apache Iceberg 0.13.1 für AWS Glue 3.0 nicht verfügbar.

In der folgenden Tabelle sind die Versionen von Iceberg aufgeführt, die in jeder AWS Glue-Version enthalten sind.

AWS Ausführung mit Glue	Unterstützte Iceberg-Version
5.0	1.7.1
4,0	1.0.0
3.0	0.13.1

Weitere Informationen zu den von AWS Glue unterstützten Data Lake-Frameworks finden Sie unterVerwendung von Data-Lake-Frameworks mit AWS Glue ETL-Jobs.

Aktivierung des Iceberg-Frameworks

Gehen Sie wie folgt vor, um Iceberg for AWS Glue zu aktivieren:

- Geben Sie `iceberg` als Wert für den Auftragsparameter `--datalake-formats` an. Weitere Informationen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.
- Erstellen Sie einen Schlüssel, der `--conf` nach Ihrem AWS Glue-Job benannt ist, und setzen Sie ihn auf den folgenden Wert. Alternativ können Sie die folgende Konfiguration mit `SparkConf` in Ihrem Skript festlegen. Diese Einstellungen helfen Apache Spark bei der korrekten Handhabung von Iceberg-Tabellen.

```
spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
--conf spark.sql.catalog.glue_catalog=org.apache.iceberg.spark.SparkCatalog
--conf spark.sql.catalog.glue_catalog.warehouse=s3://<your-warehouse-dir>/
--conf spark.sql.catalog.glue_catalog.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog
--conf spark.sql.catalog.glue_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO
```

Wenn Sie Iceberg-Tabellen lesen oder in sie schreiben, die bei Lake Formation registriert sind, folgen Sie den Anweisungen the section called “Lake Formation für FGAC” in AWS Glue 5.0 und höher. Fügen Sie in AWS Glue 4.0 die folgende Konfiguration hinzu, um die Lake Formation Formation-Unterstützung zu aktivieren.

```
--conf spark.sql.catalog.glue_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.glue_catalog.glue.id=<table-catalog-id>
```

Wenn Sie AWS Glue 3.0 mit Iceberg 0.13.1 verwenden, müssen Sie die folgenden zusätzlichen Konfigurationen festlegen, um den Amazon DynamoDB-Sperrmanager zu verwenden, um eine atomare Transaktion sicherzustellen. AWS Glue 4.0 oder höher verwendet standardmäßig optimistisches Sperren. Weitere Informationen finden Sie unter Iceberg AWS Integrations in der offiziellen Apache Iceberg-Dokumentation.

```
--conf spark.sql.catalog.glue_catalog.lock-
impl=org.apache.iceberg.aws.glue.DynamoLockManager
--conf spark.sql.catalog.glue_catalog.lock.table=<your-dynamodb-table-name>
```

Verwendung einer anderen Iceberg-Version

Um eine Version von Iceberg zu verwenden, die AWS Glue nicht unterstützt, geben Sie Ihre eigenen Iceberg-JAR-Dateien mit dem `--extra-jars` Job-Parameter an. Schließen Sie `iceberg` nicht als Wert für den Auftragsparameter `--datalake-formats` ein. Wenn Sie AWS Glue 5.0 verwenden, müssen Sie den `--user-jars-first true` Job-Parameter festlegen.

Aktivierung der Verschlüsselung für Iceberg-Tabellen

> **Note**
>
> Iceberg-Tabellen verfügen über eigene Mechanismen, um eine serverseitige Verschlüsselung zu ermöglichen. Sie sollten diese Konfiguration zusätzlich zur Sicherheitskonfiguration von AWS Glue aktivieren.

Um die serverseitige Verschlüsselung für Iceberg-Tabellen zu aktivieren, lesen Sie die Anleitung in der Iceberg-Dokumentation.

Fügen Sie die Spark-Konfiguration für Iceberg in der gesamten Region hinzu

Um eine zusätzliche Spark-Konfiguration für den regionsübergreifenden Tabellenzugriff auf Iceberg mit dem AWS Glue-Datenkatalog hinzuzufügen AWS Lake Formation, gehen Sie wie folgt vor:

1. Erstellen Sie einen Access Point mit mehreren Regionen.
2. Legen Sie die folgenden Spark-Eigenschaften fest:

```
-----
    --conf spark.sql.catalog.my_catalog.s3.use-arn-region-enabled=true \
    --conf spark.sql.catalog.{CATALOG}.s3.access-points.bucket1",
 "arn:aws:s3::<account-id>:accesspoint/<mrap-id>.mrap \
    --conf spark.sql.catalog.{CATALOG}.s3.access-points.bucket2",
 "arn:aws:s3::<account-id>:accesspoint/<mrap-id>.mrap
-----
```

Beispiel: Schreiben Sie eine Iceberg-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog

Dieses Beispielskript zeigt, wie eine Iceberg-Tabelle in Amazon S3 geschrieben wird. Das Beispiel verwendet Iceberg AWS Integrations, um die Tabelle im AWS Glue-Datenkatalog zu registrieren.

Python

```
# Example: Create an Iceberg table from a DataFrame
# and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

query = f"""
```

```
CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Scala

```
// Example: Example: Create an Iceberg table from a DataFrame
// and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

val query = """CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Alternativ können Sie mithilfe von Spark-Methoden eine Iceberg-Tabelle in Amazon S3 und den Data Catalog schreiben.

Voraussetzungen: Sie müssen einen Katalog für die Nutzung durch die Iceberg-Bibliothek bereitstellen. Wenn Sie den AWS Glue-Datenkatalog verwenden, macht AWS Glue dies einfach. Der AWS Glue-Datenkatalog ist für die Verwendung durch die Spark-Bibliotheken vorkonfiguriert als`glue_catalog`. Datenkatalogtabellen sind mit a *`databaseName`* und a *`tableName`* gekennzeichnet. Weitere Informationen zum AWS Glue-Datenkatalog finden Sie unterDatenermittlung und Katalogisierung.

Wenn Sie den AWS Glue-Datenkatalog nicht verwenden, müssen Sie einen Katalog über Spark bereitstellen APIs. Weitere Informationen finden Sie unter Spark-Konfiguration in der Iceberg-Dokumentation.

In diesem Beispiel wird mithilfe von Spark eine Iceberg-Tabelle in Amazon S3 und den Data Catalog geschrieben.

Python

```
# Example: Write an Iceberg table to S3 on the Glue Data Catalog
```

```
# Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .create()

# Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .append()
```

Scala

```
// Example: Write an Iceberg table to S3 on the Glue Data Catalog

// Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .create()

// Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .append()
```

Beispiel: Eine Iceberg-Tabelle aus Amazon S3 mithilfe des AWS Glue-Datenkatalogs lesen

In diesem Beispiel wird die Iceberg-Tabelle gelesen, die Sie in Beispiel: Schreiben Sie eine Iceberg-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog erstellt haben.

Python

Verwenden Sie für dieses Beispiel die `GlueContext.create_data_frame.from_catalog()`-Methode.

```
# Example: Read an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
```

```
df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSource-Methode.

```
// Example: Read an Iceberg table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Beispiel: Einfügen eines **`DataFrame`** in eine Iceberg-Tabelle in Amazon S3 mit dem AWS Glue Data Catalog

In diesem Beispiel werden Daten in die Iceberg-Tabelle eingefügt, die Sie in Beispiel: Schreiben Sie eine Iceberg-Tabelle in Amazon S3 und registrieren Sie sie im AWS Glue-Datenkatalog erstellt haben.

> **Note**
>
> In diesem Beispiel müssen Sie den `--enable-glue-datacatalog` Job-Parameter festlegen, um den AWS Glue-Datenkatalog als Apache Spark Hive-Metastore verwenden zu können. Weitere Informationen hierzu finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Python

Verwenden Sie für dieses Beispiel die GlueContext.write_data_frame.from_catalog()-Methode.

```
# Example: Insert into an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Verwenden Sie für dieses Beispiel die getCatalogSink-Methode.

```
// Example: Insert into an Iceberg table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Beispiel: Lesen einer Iceberg-Tabelle aus Amazon S3 mit Spark

Voraussetzungen: Sie müssen einen Katalog für die Nutzung durch die Iceberg-Bibliothek bereitstellen. Wenn Sie den AWS Glue-Datenkatalog verwenden, macht AWS Glue dies einfach. Der AWS Glue-Datenkatalog ist für die Verwendung durch die Spark-Bibliotheken vorkonfiguriert als`glue_catalog`. Datenkatalogtabellen sind mit a *`databaseName`* und a *`tableName`* gekennzeichnet. Weitere Informationen zum AWS Glue-Datenkatalog finden Sie unterDatenermittlung und Katalogisierung.

Wenn Sie den AWS Glue-Datenkatalog nicht verwenden, müssen Sie einen Katalog über Spark bereitstellen APIs. Weitere Informationen finden Sie unter Spark-Konfiguration in der Iceberg-Dokumentation.

In diesem Beispiel wird mithilfe von Spark eine Iceberg-Tabelle in Amazon S3 aus dem Data Catalog gelesen.

Python

```
# Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog

dataFrame = spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Scala

```
// Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog

val dataFrame =
 spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Beispiel: Lesen und Schreiben einer Iceberg-Tabelle mit Lake-Formation-Berechtigungskontrolle

In diesem Beispiel wird mit Lake-Formation-Berechtigungen in einer Iceberg-Tabelle gelesen und geschrieben.

> **Note**
>
> Dieses Beispiel funktioniert nur in AWS Glue 4.0. Folgen AWS Sie in Glue 5.0 und höher den Anweisungen unterthe section called “Lake Formation für FGAC”.

1. Erstellen einer Iceberg-Tabelle und Registrieren in Lake Formation.

   a. Um die Lake-Formation-Berechtigungskontrolle zu aktivieren, müssen Sie zunächst den Amazon-S3-Tabellenpfad auf Lake Formation registrieren. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Speicherorts. Sie können es entweder über die Lake Formation Formation-Konsole oder über die AWS CLI registrieren:

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-folder> --use-service-linked-role --region <REGION>
   ```

   Sobald Sie einen Amazon S3 S3-Standort registriert haben, gibt jede AWS Glue-Tabelle, die auf den Standort (oder einen seiner untergeordneten Standorte) zeigt, den Wert für den `IsRegisteredWithLakeFormation` Parameter im `GetTable` Aufruf als wahr zurück.

   b. Erstellen einer Iceberg-Tabelle, die über Spark SQL auf den registrierten Pfad verweist:

   > **Note**
   >
   > Im Folgenden sind Python-Beispiele aufgeführt.

   ```
   dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

   query = f"""
   CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
   USING iceberg
   AS SELECT * FROM tmp_<your_table_name>
   """
   spark.sql(query)
   ```

   Sie können die Tabelle auch manuell über die AWS `CreateTable` Glue-API erstellen. Weitere Informationen finden Sie unter Creating Apache Iceberg tables.

   > **Note**
   >
   > Die `UpdateTable` API unterstützt derzeit das Iceberg-Tabellenformat nicht als Eingabe für den Vorgang.

2. Erteilen Sie Lake Formation die Berechtigung für die IAM-Rolle. Sie können Berechtigungen entweder über die Lake Formation Formation-Konsole oder über die AWS CLI gewähren. Weitere

Informationen finden Sie unter: https://docs.aws.amazon.com/lake-formation/ latest/dg/granting - table-permissions.html

3. Lesen einer Iceberg-Tabelle, die bei Lake Formation registriert ist. Der Code entspricht dem Lesen einer nicht registrierten Iceberg-Tabelle. Beachten Sie, dass Ihre AWS Glue-Job-IAM-Rolle über die SELECT-Berechtigung verfügen muss, damit der Lesevorgang erfolgreich ist.

```
# Example: Read an Iceberg table from the AWS Glue Data Catalog
from awsglue.context import GlueContextfrom pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Schreiben in einer Iceberg-Tabelle, die bei Lake Formation registriert ist. Der Code entspricht dem Schreiben in eine nicht registrierte Iceberg-Tabelle. Beachten Sie, dass Ihre AWS Glue-Job-IAM-Rolle über die SUPER-Berechtigung verfügen muss, damit der Schreibvorgang erfolgreich ist.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Freigegebene Konfigurationsreferenz

Sie können die folgenden `format_options`-Werte mit jedem Formattyp verwenden.

- `attachFilename` – Eine Zeichenfolge im entsprechenden Format, die als Spaltenname verwendet werden soll. Wenn Sie diese Option angeben, wird der Name der Quelldatei für den Datensatz an den Datensatz angefügt. Der Parameterwert wird als Spaltenname verwendet.
- `attachTimestamp` – Eine Zeichenfolge im entsprechenden Format, die als Spaltenname verwendet werden soll. Wenn Sie diese Option angeben, wird die Änderungszeit der Quelldatei für den Datensatz an den Datensatz angefügt. Der Parameterwert wird als Spaltenname verwendet.

AWS Glue Data Catalog-Unterstützung für Spark-SQL-Jobs

Der AWS Glue Data Catalog ist ein Apache Hive-Metastore-kompatibler Katalog. Sie können Ihre konfigurieren AWS Glue Jobs und Entwicklungsendpunkte, um den Datenkatalog als externen Apache Hive-Metastore zu verwenden. Sie können dann direkt Apache Spark-SQL-Abfragen für die im Datenkatalog gespeicherten Tabellen ausführen. AWS Glue dynamische Frames sind standardmäßig in den Datenkatalog integriert. Mit diesem Feature können Spark-SQL-Aufträge unter Verwendung des Data Catalogs als externer Hive-Metastore gestartet werden.

Diese Funktion erfordert Netzwerkzugriff auf AWS Glue API-Endpunkt. Wählen Sie in der &Snowconsole; Ihren Auftrag aus der Tabelle. AWS Glue Für Jobs mit Verbindungen in privaten Subnetzen müssen Sie entweder einen VPC-Endpunkt oder ein NAT-Gateway konfigurieren, um den Netzwerkzugriff bereitzustellen. Informationen zur Konfiguration von VPC-Endpunkten finden Sie in Netzwerkzugriff auf Datenspeicher einrichten. Informationen zum Erstellen eines NAT-Gateways finden Sie unter NAT-Gateways im Amazon-VPC-Benutzerhandbuch.

Sie können konfigurieren AWS Glue Jobs und Entwicklungsendpunkte, indem Sie das `"--enable-glue-datacatalog": ""` Argument den Job-Argumenten bzw. den Entwicklungsendpunkt-Argumenten hinzufügen. Durch die Übergabe dieses Arguments werden bestimmte Konfigurationen in Spark festgelegt, die ihm Zugriff auf den Data Catalog als externer Hive-Metastore ermöglichen. Es aktiviert auch die Hive-Unterstützung für das `SparkSession` Objekt, das in der erstellt wurde AWS Glue Job- oder Entwicklungsendpunkt.

Um den Datenkatalogzugriff zu aktivieren, aktivieren Sie das Kontrollkästchen AWS Glue-Datenkatalog als Hive-Metastore verwenden in der Optionsgruppe Katalog auf der Seite Job hinzufügen oder Endpunkt hinzufügen in der Konsole. Beachten Sie, dass die für den Auftrag oder den Entwicklungsendpunkt verwendete IAM-Rolle über `glue:CreateDatabase`-Berechtigungen verfügen sollte. Sofern nicht bereits vorhanden, wird im Data Catalog eine Datenbank namens „`default`“ angelegt.

Schauen Sie sich ein Beispiel dazu an, wie Sie dieses Feature in Ihren Spark-SQL-Aufträgen nutzen können. Im folgenden Beispiel wird davon ausgegangen, dass Sie das US-Gesetzgeber-Dataset durchsucht haben, das unter `s3://awsglue-datasets/examples/us-legislators` verfügbar ist.

Um Daten aus den im AWS Glue-Datenkatalog definierten Tabellen zu serialisieren/deserialisieren, benötigt Spark SQL die SerDeHive-Klasse für das Format, das im AWS Glue-Datenkatalog im Klassenpfad des Spark-Jobs definiert ist.

SerDes denn bestimmte gängige Formate werden vertrieben von AWS Glue. Im Folgenden finden Sie die Amazon S3 S3-Links für diese:

- JSON
- XML
- Grok

Fügen Sie das JSON SerDe als zusätzliches JAR zum Entwicklungsendpunkt hinzu. Bei Jobs können Sie das SerDe mithilfe des `--extra-jars` Arguments im Argumentfeld hinzufügen. Weitere Informationen finden Sie unter Verwenden von Auftragsparametern in AWS Glue-Jobs.

Im Folgenden finden Sie ein Beispiel für eine JSON-Eingabe zum Erstellen eines Entwicklungsendpunkts mit aktiviertem Data Catalog für Spark SQL.

```
{
    "EndpointName": "Name",
    "RoleArn": "role_ARN",
    "PublicKey": "public_key_contents",
    "NumberOfNodes": 2,
    "Arguments": {
      "--enable-glue-datacatalog": ""
    },
    "ExtraJarsS3Path": "s3://crawler-public/json/serde/json-serde.jar"
}
```

Führen Sie nun eine Abfrage der aus dem USA-Gesetzgeber-Dataset erstellten Tabellen mit Spark SQL durch.

```
>>> spark.sql("use legislators")
DataFrame[]
>>> spark.sql("show tables").show()
+-----------+-----------------+-----------+
|   database|        tableName|isTemporary|
+-----------+-----------------+-----------+
|legislators|       areas_json|      false|
|legislators|   countries_json|      false|
|legislators|      events_json|      false|
|legislators| memberships_json|      false|
```

```
|legislators|organizations_json|      false|
|legislators|      persons_json|      false|
+-----------+------------------+-----------+
>>> spark.sql("describe memberships_json").show()
+--------------------+---------+-----------------+
|            col_name|data_type|          comment|
+--------------------+---------+-----------------+
|             area_id|   string|from deserializer|
|     on_behalf_of_id|   string|from deserializer|
|     organization_id|   string|from deserializer|
|                role|   string|from deserializer|
|           person_id|   string|from deserializer|
|legislative_perio...|   string|from deserializer|
|          start_date|   string|from deserializer|
|            end_date|   string|from deserializer|
+--------------------+---------+-----------------+
```

Wenn die SerDe Klasse für das Format im Klassenpfad des Jobs nicht verfügbar ist, wird eine Fehlermeldung ähnlich der folgenden angezeigt.

```
>>> spark.sql("describe memberships_json").show()

Caused by: MetaException(message:java.lang.ClassNotFoundException Class
 org.openx.data.jsonserde.JsonSerDe not found)
    at
 org.apache.hadoop.hive.metastore.MetaStoreUtils.getDeserializer(MetaStoreUtils.java:399)
    at
 org.apache.hadoop.hive.ql.metadata.Table.getDeserializerFromMetaStore(Table.java:276)
    ... 64 more
```

Um nur die eindeutigen `organization_id`s aus der `memberships`-Tabelle anzuzeigen, führen Sie die folgende SQL-Abfrage durch.

```
>>> spark.sql("select distinct organization_id from memberships_json").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Wenn Sie bei Dynamic Frames genauso verfahren müssen, führen Sie die folgenden Schritte aus.

```
>>> memberships = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="memberships_json")
>>> memberships.toDF().createOrReplaceTempView("memberships")
>>> spark.sql("select distinct organization_id from memberships").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Obwohl sie für ETL-Operationen optimiert DynamicFrames sind, bietet die Möglichkeit, Spark SQL direkt auf den Datenkatalog zuzugreifen, eine übersichtliche Möglichkeit, komplexe SQL-Anweisungen auszuführen oder bestehende Anwendungen zu portieren.

Verwenden von Auftragslesezeichen

AWS Glue for Spark verwendet Job-Lesezeichen, um Daten zu verfolgen, die bereits verarbeitet wurden. Eine Übersicht über das Feature Auftragslesezeichen und was es unterstützt, finden Sie unter the section called “Verfolgen von verarbeiteten Daten mit Auftragslesezeichen”. Wenn Sie einen AWS Glue Job mit Lesezeichen programmieren, haben Sie Zugriff auf Flexibilität, die bei visuellen Aufträgen nicht verfügbar ist.

- Wenn Sie aus JDBC lesen, können Sie die Spalte (n) angeben, die als Lesezeichenschlüssel in Ihrem Skript verwendet werden sollen. AWS Glue
- Sie können auswählen, welches `transformation_ctx` bei jedem Methodenaufruf angewendet werden soll.

Rufen Sie *`job.init`* immer am Anfang des Skripts und am Ende des *`job.commit`* Skripts mit entsprechend konfigurierten Parametern auf. Diese beiden Funktionen initialisieren den Lesezeichen-Service und aktualisieren so die Statusänderung des Services. Lesezeichen funktionieren nicht, ohne sie anzurufen.

Lesezeichenschlüssel angeben

Bei JDBC-Workflows merkt sich das Lesezeichen, welche Zeilen Ihr Auftrag gelesen hat, indem es die Werte der Schlüsselfelder mit einem Wert aus dem Lesezeichen vergleicht. Dies ist für Amazon-S3-Workflows nicht erforderlich oder anwendbar. Wenn Sie ein AWS Glue Skript ohne den visuellen Editor schreiben, können Sie angeben, welche Spalte mit Lesezeichen verfolgt werden soll. Sie

können auch mehrere Spalten festlegen. Bei der Angabe benutzerdefinierter Lesezeichenschlüssel sind Lücken in der Wertefolge zulässig.

> **Warning**
>
> Wenn benutzerdefinierte Lesezeichenschlüssel verwendet werden, müssen diese jeweils streng monoton ansteigend oder abfallend sein. Wenn Sie zusätzliche Felder für einen zusammengesetzten Schlüssel auswählen, erfüllen Felder für Konzepte wie „Nebenversionen“ oder „Revisionsnummern“ dieses Kriterium nicht, da ihre Werte im gesamten Datensatz wiederverwendet werden.

Sie können `jobBookmarkKeys` und `jobBookmarkKeysSortOrder` auf folgende Weise angeben:

- `create_dynamic_frame.from_catalog` – verwenden Sie `additional_options`.
- `create_dynamic_frame.from_options` – verwenden Sie `connection_options`.

Transformationskontext

Viele der AWS Glue PySpark Dynamische Frame-Methoden enthalten einen optionalen Parameter namens`transformation_ctx`, der eine eindeutige Kennung für die ETL-Operatorinstanz darstellt. Der Parameter `transformation_ctx` wird verwendet, um Zustandsinformationen innerhalb eines Auftragslesezeichens für den gegebenen Operator zu identifizieren. Konkret AWS Glue verwendet`transformation_ctx`, um den Schlüssel für den Lesezeichenstatus zu indizieren.

> **Warning**
>
> `transformation_ctx` dient als Schlüssel, um den Lesezeichenstatus nach einer bestimmten Quelle in Ihrem Skript zu durchsuchen. Damit das Lesezeichen ordnungsgemäß funktioniert, sollten Sie stets die Konsistenz von Quelle und `transformation_ctx` wahren. Durch Ändern der Quelleigenschaft oder Umbenennen von `transformation_ctx` kann das vorherige Lesezeichen ungültig werden und die zeitstempelbasierte Filterung führt möglicherweise nicht zum richtigen Ergebnis.

Damit Auftragslesezeichen richtig funktionieren, aktivieren Sie den Parameter für Auftragslesezeichen und setzen Sie den Parameter `transformation_ctx`. Wenn Sie den `transformation_ctx`-Parameter nicht übergeben, sind keine Auftragslesezeichen für einen in der Methode

verwendeten dynamischen Frame bzw. eine dort verwendete Tabelle aktiviert. Bei einem ETL-Auftrag beispielsweise, der zwei Amazon-S3-Quellen liest und verbindet, können Sie den `transformation_ctx`-Parameter auch nur an jene Methoden übergeben, die Sie für Lesezeichen aktivieren wollen. Wenn Sie das Auftragslesezeichen für einen Auftrag zurücksetzen, werden alle Transformationen, die mit dem Auftrag verbunden sind, unabhängig vom verwendeten `transformation_ctx` zurückgesetzt.

Weitere Informationen über die `DynamicFrameReader`-Klasse finden Sie unter DynamicFrameReader Klasse. Weitere Hinweise zu PySpark Erweiterungen finden Sie unterAWS Referenz für PySpark Klebererweiterungen.

Beispiele

Example

Im Folgenden finden Sie ein Beispiel für ein generiertes Skript für eine Amazon-S3-Datenquelle. Die Teile des Skripts, die für die Nutzung von Auftragslesezeichen erforderlich sind, werden kursiv dargestellt. Weitere Informationen zu diesen Elementen finden Sie in der GlueContext Klasse-API und in der DynamicFrameWriter Klasse-API.

```
# Sample Script
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "database",
    table_name = "relatedqueries_csv",
    transformation_ctx = "datasource0"
)

applymapping1 = ApplyMapping.apply(
```

```
    frame = datasource0,
    mappings = [("col0", "string", "name", "string"), ("col1", "string", "number",
 "string")],
    transformation_ctx = "applymapping1"
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://input_path"},
    format = "json",
    transformation_ctx = "datasink2"
)


job.commit()
```

Example

Im Folgenden finden Sie ein Beispiel für ein generiertes Skript für eine JDBC-Quelle. Die Quelltabelle ist eine Mitarbeitertabelle mit der empno-Spalte als Primärschlüssel. Der Auftrag verwendet standardmäßig einen sequenziellen Primärschlüssel als Lesezeichenschlüssel, wenn kein Lesezeichenschlüssel angegeben ist. Da empno nicht notwendigerweise sequentiell ist, kann es Lücken in den Werten geben – es gilt nicht als Standardschlüssel für Lesezeichen. Daher wird das Skript explizit empno als Lesezeichenschlüssel bezeichnet. Dieser Teil des Codes ist kursiv dargestellt.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "hr",
    table_name = "emp",
    transformation_ctx = "datasource0",
    additional_options = {"jobBookmarkKeys":["empno"],"jobBookmarkKeysSortOrder":"asc"}
)

applymapping1 = ApplyMapping.apply(
    frame = datasource0,
    mappings = [("ename", "string", "ename", "string"), ("hrly_rate", "decimal(38,0)",
 "hrly_rate", "decimal(38,0)"), ("comm", "decimal(7,2)", "comm", "decimal(7,2)"),
 ("hiredate", "timestamp", "hiredate", "timestamp"), ("empno", "decimal(5,0)", "empno",
 "decimal(5,0)"), ("mgr", "decimal(5,0)", "mgr", "decimal(5,0)"), ("photo", "string",
 "photo", "string"), ("job", "string", "job", "string"), ("deptno", "decimal(3,0)",
 "deptno", "decimal(3,0)"), ("ssn", "decimal(9,0)", "ssn", "decimal(9,0)"), ("sal",
 "decimal(7,2)", "sal", "decimal(7,2)")],
    transformation_ctx = "applymapping1"
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://hr/employees"},
    format = "csv",
    transformation_ctx = "datasink2"
)

job.commit()
```

Erkennung sensibler Daten außerhalb von AWS Glue Studio verwenden

AWS Glue In Studio können Sie sensible Daten erkennen. Sie können die Funktion zur Erkennung sensibler Daten jedoch auch außerhalb von AWS Glue Studio verwenden.

Eine vollständige Liste der verwalteten vertraulichen Datentypen finden Sie unter Verwaltete Datentypen.

Erkennung sensibler Daten mithilfe AWS verwalteter PII-Typen

AWS Glue stellt zwei APIs in einem AWS Glue ETL-Job zur Verfügung. Diese sind `detect()` und `classifyColumns()`:

```
 detect(frame: DynamicFrame,
     entityTypesToDetect: Seq[String],
     outputColumnName: String = "DetectedEntities",
     detectionSensitivity: String = "LOW"): DynamicFrame

detect(frame: DynamicFrame,
     detectionParameters: JsonOptions,
     outputColumnName: String = "DetectedEntities",
     detectionSensitivity: String = "LOW"): DynamicFrame

 classifyColumns(frame: DynamicFrame,
     entityTypesToDetect: Seq[String],
     sampleFraction: Double = 0.1,
     thresholdFraction: Double = 0.1,
     detectionSensitivity: String = "LOW")
```

Sie können die `detect()` API verwenden, um AWS verwaltete PII-Typen und benutzerdefinierte Entitätstypen zu identifizieren. Es wird automatisch eine neue Spalte mit dem Erkennungsergebnis erstellt. Die `classifyColumns()` API gibt eine Map zurück, in der Schlüssel Spaltennamen und Werte eine Liste der erkannten Entitätstypen sind. `SampleFraction` gibt den Bruchteil der Daten an, der beim Scannen nach PII-Entitäten geprobt werden muss und `ThresholdFraction` gibt den Bruchteil der Daten an, der erfüllt sein muss, damit eine Spalte als PII-Daten identifiziert werden kann.

Erkennung auf Zeilenebene

Im Beispiel führt der Job mithilfe von `detect()` und `classifyColumns()` APIs die folgenden Aktionen aus:

- liest Daten aus einem Amazon S3 Bucket und wandelt sie in einen DynamicFrame um
- erkennt Instances von „E-Mail“ und „Kreditkarte“ im DynamicFrame
- gibt einen DynamicFrame mit Originalwerten plus einer Spalte zurück, die das Erkennungsergebnis für jede Zeile umfasst
- den zurückgegebenen DynamicFrame in einen anderen Pfad schreiben Amazon S3

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
```

```
 import com.amazonaws.services.glue.util.GlueArgParser
 import com.amazonaws.services.glue.util.Job
 import com.amazonaws.services.glue.util.JsonOptions
 import org.apache.spark.SparkContext
 import scala.collection.JavaConverters._
 import com.amazonaws.services.glue.ml.EntityDetector

 object GlueApp {
   def main(sysArgs: Array[String]) {
     val spark: SparkContext = new SparkContext()
     val glueContext: GlueContext = new GlueContext(spark)
     val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
     Job.init(args("JOB_NAME"), glueContext, args.asJava)
     val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

     val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD"))

     glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

     Job.commit()
   }
 }
```

Erkennung auf Zeilenebene mit fein abgestuften Aktionen

In dem Beispiel führt der Job die folgenden Aktionen mit dem aus: `detect()` APIs

- liest Daten aus einem Amazon-S3-Bucket und wandelt sie in einen DynamicFrame um
- Erkennung sensibler Datentypen für „USA_PTIN“, „BANK_ACCOUNT“, „USA_SSN“, „USA_PASSPORT_NUMBER“ und „PHONE_NUMBER“ im DynamicFrame
- gibt einen DynamicFrame mit bearbeiteten maskierten Werten plus einer Spalte zurück, die das Erkennungsergebnis für jede Zeile umfasst
- schreibt den zurückgegebenen DynamicFrame in einen anderen Amazon-S3-Pfad

Im Gegensatz zur obigen API werden hier detaillierte Aktionen für die Erkennung von Entitätstypen verwendet. `detect()` Weitere Informationen finden Sie unter Erkennungsparameter für die Verwendung von `detect()`.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
 "withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
 options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
 transformationContext="AmazonS3_node_source").getDynamicFrame()

    val detectionParameters = JsonOptions(
      """
        {
          "USA_DRIVING_LICENSE": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Driving License"],
            "actionOptions": {
              "matchPattern": "[0-9]",
              "redactChar": "*"
            }
          }],
          "BANK_ACCOUNT": [{
            "action": "DETECT",
            "sourceColumns": ["*"]
          }],
          "USA_SSN": [{
            "action": "SHA256_HASH",
```

```
            "sourceColumns": ["SSN"]
          }],
          "IP_ADDRESS": [{
            "action": "REDACT",
            "sourceColumns": ["IP Address"],
            "actionOptions": {"redactText": "*****"}
          }],
          "PHONE_NUMBER": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Phone Number"],
            "actionOptions": {
              "numLeftCharsToExclude": 1,
              "numRightCharsToExclude": 0,
              "redactChar": "*"
            }
          }]
        }
      """
    )

    val frameWithDetectedPII = EntityDetector.detect(frame, detectionParameters,
 "DetectedEntities", "HIGH")

    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput/", "partitionKeys": []}"""),
 transformationContext="AmazonS3_node_target",
 format="json").writeDynamicFrame(frameWithDetectedPII)

    Job.commit()
  }
}
```

Erkennung auf Spaltenebene

In dem Beispiel führt der Job die folgenden Aktionen mit dem aus `classifyColumns()`APIs:

- liest Daten aus einem Amazon-S3-Bucket und wandelt sie in einen DynamicFrame um
- erkennt Instances von „E-Mail“ und „Kreditkarte“ im DynamicFrame

- Stellen Sie die Parameter so ein, dass 100 % der Spalte erfasst werden, markieren Sie eine Entität als erkannt, wenn sie sich in 10 % der Zellen befindet, und stellen Sie die Empfindlichkeit „LOW“ ein.
- Gibt eine Zuordnung zurück, in der Spaltennamen die Schlüssel sind und die Liste der erkannten Entitätstypen die Werte bildet.
- schreibt den zurückgegebenen DynamicFrame in einen anderen Amazon-S3-Pfad

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.DynamicFrame
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
 "\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
 connectionType="s3", format="csv", options=JsonOptions("""{"paths": ["s3://
pathToSource"], "recurse": true}"""), transformationContext="frame").getDynamicFrame()

    import glueContext.sparkSession.implicits._

    val detectedDataFrame = EntityDetector.classifyColumns(
        frame,
        entityTypesToDetect = Seq("CREDIT_CARD", "PHONE_NUMBER"),
        sampleFraction = 1.0,
        thresholdFraction = 0.1,
        detectionSensitivity = "LOW"
    )
    val detectedDF = (detectedDataFrame).toSeq.toDF("columnName", "entityTypes")
    val DetectSensitiveData_node = DynamicFrame(detectedDF, glueContext)
```

```
    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput", "partitionKeys": []}"""), transformationContext="someCtx",
 format="json").writeDynamicFrame(DetectSensitiveData_node)

    Job.commit()
  }
}
```

Erkennung sensibler Daten mithilfe von AWS CustomEntityType PII-Typen

Sie können benutzerdefinierte Entitäten über AWS Studio definieren. Um diese Funktion jedoch von AWS Studio aus verwenden zu können, müssen Sie zuerst die benutzerdefinierten Entitätstypen definieren und dann die definierten benutzerdefinierten Entitätstypen zur Liste von hinzufügen`entityTypesToDetect`.

Wenn Ihre Daten bestimmte sensible Datentypen enthalten (z. B. „Mitarbeiter-ID“), können Sie benutzerdefinierte Entitäten erstellen, indem Sie die API `CreateCustomEntityType()` aufrufen. Das folgende Beispiel definiert den benutzerdefinierten Entitätstyp 'EMPLOYEE_ID' für die API `CreateCustomEntityType()` mit den Anforderungsparametern:

```
{
    "name": "EMPLOYEE_ID",
    "regexString": "\d{4}-\d{3}",
    "contextWords": ["employee"]
}
```

Ändern Sie dann den Auftrag, um den neuen benutzerdefinierten vertraulichen Datentyp zu verwenden, indem Sie der API `EntityDetector()` den benutzerdefinierten Entitätstyp (EMPLOYEE_ID) hinzufügen:

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
```

```
 import com.amazonaws.services.glue.util.JsonOptions
 import org.apache.spark.SparkContext
 import scala.collection.JavaConverters._
 import com.amazonaws.services.glue.ml.EntityDetector

 object GlueApp {
   def main(sysArgs: Array[String]) {
     val spark: SparkContext = new SparkContext()
     val glueContext: GlueContext = new GlueContext(spark)
     val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
     Job.init(args("JOB_NAME"), glueContext, args.asJava)
     val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

     val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD", "EMPLOYEE_ID"))

     glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

     Job.commit()
   }
 }
```

> **Note**
>
> Wenn ein benutzerdefinierter sensibler Datentyp mit demselben Namen wie ein vorhandener verwalteter Entitätstyp definiert ist, hat der benutzerdefinierte sensible Datentyp Vorrang und überschreibt die Logik des verwalteten Entitätstyps.

Erkennungsparameter für die Verwendung von **`detect()`**

Diese Methode wird zum Erkennen von Entitäten in a verwendet DynamicFrame. Es gibt eine neue Spalte DataFrame mit Originalwerten und einer zusätzlichen Spalte zurück outputColumnName , die Metadaten zur PII-Erkennung enthält. Eine benutzerdefinierte Maskierung kann durchgeführt werden,

nachdem DynamicFrame dies innerhalb der AWS Glue Stattdessen kann das Skript oder die Detect () -API mit feinkörnigen Aktionen verwendet werden.

```
detect(frame: DynamicFrame,
       entityTypesToDetect: Seq[String],
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Parameter:

- frame — (Typ:`DynamicFrame`) Die Eingabe, die die zu verarbeitenden Daten DynamicFrame enthält.
- entityTypesToDetect — (Typ:`[Seq[String]`) Liste der zu erkennenden Entitätstypen. Dabei kann es sich entweder um verwaltete Entitätstypen oder um benutzerdefinierte Entitätstypen handeln.
- outputColumnName— (Typ:`String`, Standard: "DetectedEntities„) Der Name der Spalte, in der erkannte Entitäten gespeichert werden. Falls nicht angegeben, ist der Standardspaltenname "DetectedEntities“.
- detectionSensitivity – (Typ:`String`, Optionen: „LOW“ oder „HIGH“, Standard: „LOW“) Gibt die Empfindlichkeit des Erkennungsprozesses an. Gültige Optionen sind „LOW“ oder „HIGH“. Falls nicht angegeben, ist die Standardempfindlichkeit auf „LOW“ eingestellt.

`outputColumnName`-Einstellungen:

Der Name der Spalte, in der erkannte Entitäten gespeichert werden sollen. Falls nicht angegeben, lautet der Standardspaltenname "DetectedEntities“. Für jede Zeile in der Ausgabespalte enthält die zusätzliche Spalte eine Zuordnung des Spaltennamens zu den Metadaten der erkannten Entität mit den folgenden Schlüssel-Wert-Paaren:

- entityType – Der erkannte Entitätstyp.
- start – Die Startposition der erkannten Entität in den Originaldaten.
- end – Die Endposition der erkannten Entität in den Originaldaten.
- actionUsed — Die Aktion, die an der erkannten Entität ausgeführt wurde (z. B. „DETECT“, „REDACT“, „PARTIAL_REDACT“, "_HASH“). SHA256

Beispiel:

```
{
   "DetectedEntities":{
      "SSN Col":[
         {
            "entityType":"USA_SSN",
            "actionUsed":"DETECT",
            "start":4,
            "end":15
         }
      ],
      "Random Data col":[
         {
            "entityType":"BANK_ACCOUNT",
            "actionUsed":"PARTIAL_REDACT",
            "start":4,
            "end":13
         },
         {
            "entityType":"IP_ADDRESS",
            "actionUsed":"REDACT",
            "start":4,
            "end":13
         }
      ]
   }
}
```

Erkennungsparameter für **`detect()`** mit detaillierten Aktionen

Diese Methode wird verwendet, um Entitäten in einer unter Verwendung bestimmter Parameter zu erkennen. DynamicFrame Sie gibt eine neue Spalte zurück, DataFrame bei der die Originalwerte durch maskierte vertrauliche Daten ersetzt wurden`outputColumnName`, und eine zusätzliche Spalte mit Metadaten zur PII-Erkennung.

```
detect(frame: DynamicFrame,
       detectionParameters: JsonOptions,
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Parameter:

- frame — (Typ:`DynamicFrame`): Die Eingabe, die die zu verarbeitenden Daten DynamicFrame enthält.
- detectionParameters – (Typ: `Json0ptions`): JSON-Optionen, die Parameter für den Erkennungsprozess angeben.
- outputColumnName— (type:`String`, default: "DetectedEntities„): Der Name der Spalte, in der erkannte Entitäten gespeichert werden. Falls nicht angegeben, ist der Standardspaltenname "DetectedEntities“.
- detectionSensitivity – (Typ:`String`, Optionen: „LOW“ oder „HIGH“, Standard: „LOW“): Gibt die Empfindlichkeit des Erkennungsprozesses an. Gültige Optionen sind „LOW“ oder „HIGH“. Falls nicht angegeben, ist die Standardempfindlichkeit auf „LOW“ eingestellt.

`detectionParameters`-Einstellungen

Wenn keine Einstellungen enthalten sind, werden Standardwerte verwendet.

- action — (Typ:`String`, Optionen: „DETECT“, „REDACT“, „PARTIAL_REDACT“, "SHA256_HASH“) Gibt die Aktion an, die an der Entität ausgeführt werden soll. Erforderlich Beachten Sie, dass Aktionen, die eine Maskierung durchführen (alle außer „DETECT“), nur einmal pro Spalte ausgeführt werden. Dies ist eine vorbeugende Maßnahme zur Maskierung zusammengeführter Entitäten.
- sourceColumns – (Typ:`List[String]`, Standard: [„*“]) Liste der Quellspaltennamen, anhand derer die Entität erkannt werden soll. Wenn hier nichts angegeben ist, lautet die Standardeinstellung [„*“]. `IllegalArgumentException` wird ausgelöst, wenn ein ungültiger Spaltenname verwendet wird.
- sourceColumnsToAusschließen — (Typ:`List[String]`) Liste der Quellspaltennamen, anhand derer die Entität erkannt werden soll. Verwenden Sie entweder `sourceColumns` oder `sourceColumnsToExclude`. `IllegalArgumentException` wird ausgelöst, wenn ein ungültiger Spaltenname verwendet wird.
- actionOptions – Zusätzliche Optionen, die auf der angegebenen Aktion basieren:
  - Für „DETECT“ und "SHA256_HASH“ sind keine Optionen zulässig.
  - Für „REDACT“:
    - redactText – (Typ:`String`, Standard: „*****“) Text, der die erkannte Entität ersetzen soll.
  - Für „PARTIAL_REDACT“:
    - redactChar – (Typ:`String`, Standard: „*“) Zeichen, das jedes erkannte Zeichen in der Entität ersetzen soll.

- matchPattern – (Typ:`String`) Regex-Muster für teilweises Redigieren. Kann nicht mit numLeftChars ToExclude oder `numRightCharsToExclude` kombiniert werden.
- numLeftCharsToExclude— (Typ:`String, integer`) Anzahl der verbleibenden Zeichen, die ausgeschlossen werden sollen. Kann nicht mit matchPattern kombiniert werden, kann aber mit `numRightCharsToExclude` verwendet werden.
- numRightCharsToExclude— (Typ:`String, integer`) Anzahl der auszuschließenden rechten Zeichen. Kann nicht mit matchPattern kombiniert werden, kann aber mit `numRightCharsToExclude` verwendet werden.

`outputColumnName`-Einstellungen

Siehe outputColumnName Einstellungen

Erkennungsparameter für **`classifyColumns()`**

Diese Methode wird zum Erkennen von Entitäten in a verwendet DynamicFrame. Gibt eine Zuordnung zurück, in der Spaltennamen die Schlüssel sind und die Liste der erkannten Entitätstypen die Werte bildet. Eine benutzerdefinierte Maskierung kann durchgeführt werden, nachdem dies innerhalb der zurückgegeben wurde AWS Glue skript.

```
classifyColumns(frame: DynamicFrame,
                entityTypesToDetect: Seq[String],
                sampleFraction: Double = 0.1,
                thresholdFraction: Double = 0.1,
                detectionSensitivity: String = "LOW")
```

Parameter:

- frame — (Typ:`DynamicFrame`) Die Eingabe, die die zu verarbeitenden Daten DynamicFrame enthält.
- entityTypesToDetect — (Typ:`Seq[String]`) Liste der zu erkennenden Entitätstypen. Dabei kann es sich entweder um verwaltete Entitätstypen oder um benutzerdefinierte Entitätstypen handeln.
- sampleFraction – (Typ:`Double`, Standard: 10 %) Der Bruchteil der Daten, die beim Scannen nach PII-Entitäten erfasst werden sollen.
- thresholdFraction – (Typ:`Double`, Standard: 10 %): Der Bruchteil der Daten, bei dem die Bedingung erfüllt sein muss, damit eine Spalte als PII-Daten identifiziert werden kann.

- detectionSensitivity – (Typ:`String`, Optionen: „LOW“ oder „HIGH“, Standard: „LOW“) Gibt die Empfindlichkeit des Erkennungsprozesses an. Gültige Optionen sind „LOW“ oder „HIGH“. Falls nicht angegeben, ist die Standardempfindlichkeit auf „LOW“ eingestellt.

Verwaltete sensible Datentypen

Globale Einheiten

Datentyp	Kategorie	Beschreibung
PERSON_NAME	Universal	Der Name der Person.
EMAIL	Persönlich	Die E-Mail-Adresse.
IP_ADDRESS	Computer	Die IP-Adresse
MAC_ADRESS	Persönlich	Die MAC-Adresse.

US-Datentypen

Datentyp	Beschreibung
BANK_ACCOUNT	Die Bankkontonummer. Nicht spezifisch für ein Land oder eine Region, es werden jedoch nur US-amerikanische und kanadische Kontoform ate erkannt.
CREDIT_CARD	Die Kreditkartennummer.
PHONE_NUMBER	Die Telefonnummer. Nicht spezifisch für ein Land oder eine Region, jedoch werden derzeit nur US-amerikanische und kanadische Telefonnummern erkannt.
USA_ATIN	Die vom Internal Revenue Service ausgestellte US-Adoptionssteuer-Identifikationsnummer.

Datentyp	Beschreibung
USA_CPT_CODE	Der CPT-Code (US-spezifisch).
USA_DEA_NUMBER	Die DEA-Nummer (US-spezifisch).
USA_DRIVING_LICENSE	Die Führerscheinnummer (US-spezifisch).
USA_HCPCS_CODE	Der HCPCS-Code (US-spezifisch).
USA_HEALTH_INSURANCE_CLAIM_NUMBER	Health Insurance Claim Number (US-spezifisch).
USA_ITIN	Die ITIN (für US-amerikanische Personen oder Entitäten).
USA_MEDICARE_BENEFICIARY_IDENTIFIER	Medicare-Versichertennummer (US-spezifisch).
USA_NATIONAL_DRUG_CODE	Der NDC-Code (US-spezifisch).
USA_NATIONAL_PROVIDER_IDENTIFIER	Die National Provider Identifier Nummer (US-spezifisch).
USA_PASSPORT_NUMBER	Die Passnummer (für US-Personen).
USA_PTIN	Die vom Internal Revenue Service ausgegebene Steueridentifikationsnummer des US-Erstellers.
USA_SSN	Die Sozialversicherungsnummer (für US-Personen).

Datentypen für Argentinien

Datentyp	Beschreibung
ARGENTINA_TAX_IDENTIFICATION_NUMBER	Argentinische Steueridentifikationsnummer. Auch bekannt als CUIT oder CUIL.

Datentypen für Australien

Datentyp	Beschreibung
AUSTRALIA_BUSINESS_NUMBER	Australische Unternehmensnummer (ABN). Eine vom Australian Business Register (ABR) ausgestellte eindeutige Kennung zur Identifizierung von Unternehmen gegenüber der Regierung und der Gemeinde.
AUSTRALIA_COMPANY_NUMBER	Australische Firmennummer (ACN). Eindeutige Kennung, die von der Australian Securities and Investments Commission ausgestellt wird.
AUSTRALIA_DRIVING_LICENSE	Eine Führerscheinnummer für Australien.
AUSTRALIA_MEDICARE_NUMBER	Australische Medicare-Nummer. Persönliche Kennung, die von der Australian Health Insurance Commission ausgestellt wird.
AUSTRALIA_PASSPORT_NUMBER	Australische Reisepassnummer.
AUSTRALIA_TAX_FILE_NUMBER	Australische Steuernummer (TFN). Wird von der australischen Steuerbehörde (Australian Taxation Office, ATO) an Steuerzahler (Einzelpersonen, Unternehmen usw.) für Steuergeschäfte ausgestellt.

Datentypen für Österreich

Datentyp	Beschreibung
AUSTRIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Österreich).
AUSTRIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Österreich).

Datentyp	Beschreibung
AUSTRIA_SSN	Die Sozialversicherungsnummer (für Personen aus Österreich).
AUSTRIA_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Österreich).
AUSTRIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Österreich).

Datentypen für Balkanländer

Datentyp	Beschreibung
BOSNIA_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für Bürger von Bosnien-Herzegowina.
KOSOVO_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für Kosovo.
MACEDONIA_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer für Mazedonien.
MONTENEGRO_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für Montenegro.
SERBIA_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für Serbien.
SERBIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Serbien).
VOJVODINA_UNIQUE_MASTER_CITIZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für die Vojvodina.

Belgische Datentypen

Datentyp	Beschreibung
BELGIUM_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Belgien).
BELGIUM_NATIONAL_IDENTIFICATION_NUMBER	Die belgische Nationalregisternummer (BNN).
BELGIUM_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Belgien).
BELGIUM_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Belgien).
BELGIUM_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Belgien).

Datentypen für Brasilien

Datentyp	Beschreibung
BRAZIL_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für Brasilien).
BRAZIL_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Brasilien).
BRAZIL_NATIONAL_REGISTRY_OF_LEGAL_ENTITIES_NUMBER	Die an Unternehmen ausgegebene Identifikationsnummer (spezifisch für Brasilien), auch bekannt als CNPJ.
BRAZIL_NATURAL_PERSON_REGISTRY_NUMBER	Registrierungsnummer natürlicher Personen, auch bekannt als CPF.

Datentypen in Bulgarien

Datentyp	Beschreibung
BULGARIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Bulgarien).
BULGARIA_UNIFORM_CIVIL_NUMBER	Einheitliche Zivilnummer (EGN), die als nationale Identifikationsnummer dient.
BULGARIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Bulgarien).

Datentypen für Kanada

Datentyp	Beschreibung
CANADA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Kanada).
CANADA_GOVERNMENT_IDENTIFICATION_CARD_NUMBER	Die nationale Kennung (spezifisch für Kanada).
CANADA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Kanada).
CANADA_PERMANENT_RESIDENCE_NUMBER	Nummer des ständigen Wohnsitzes (PR-Kartennummer).
CANADA_PERSONAL_HEALTH_NUMBER	Die eindeutige Kennung für das Gesundheitswesen (PHN-Nummer).
CANADA_SOCIAL_INSURANCE_NUMBER	Die kanadische Sozialversicherungsnummer (SIN).

Datentypen in Chile

Datentyp	Beschreibung
CHILE_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Chile).

Datentyp	Beschreibung
CHILE_NATIONAL_IDENTIFICATION_NUMBER	Die chilenische Landeskennung, auch bekannt als RUT oder RUN.

Datentypen für China, Hongkong, Macao und Taiwan

Datentyp	Beschreibung
CHINA_IDENTIFICATION	Die chinesische Kennung.
CHINA_LICENSE_PLATE_NUMBER	Die Führerscheinnummer (spezifisch für China).
CHINA_MAINLAND_TRAVEL_PERMIT_ID_HONG_KONG_MACAU	Die Reisegenehmigung für das Festland für Einwohner von Hongkong und Macau.
CHINA_MAINLAND_TRAVEL_PERMIT_ID_TAIWAN	Die von der Regierung der Volksrepublik China (VR China) ausgestellte Reisegenehmigung für das Festland für Einwohner Taiwans.
CHINA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für China).
CHINA_PHONE_NUMBER	Die Telefonnummer (spezifisch für China).
HONG_KONG_IDENTITY_CARD	Das offizielle Ausweisdokument, ausgestellt von der Einwanderungsbehörde von Hongkong.
MACAU_RESIDENT_IDENTITY_CARD	Die Macau Resident Identity Card oder BIR ist ein offizieller Personalausweis, der vom Identification Services Bureau von Macau ausgestellt wird.
TAIWAN_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Taiwan).
TAIWAN_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Taiwan).

Datentypen für Kolumbien

Datentyp	Beschreibung
COLOMBIA_PERSONAL_IDENTIFICATION_NUMBER	Eindeutige Kennung, die Kolumbianern bei der Geburt zugewiesen wird.
COLOMBIA_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Kolumbien).

Datentypen für Kroatien

Datentyp	Beschreibung
CROATIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Kroatien).
CROATIA_IDENTITY_NUMBER	Die nationale Kennung (spezifisch für Kroatien).
CROATIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Kroatien).
CROATIA_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (OIB).

Datentypen für Zypern

Datentyp	Beschreibung
CYPRUS_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Zypern).
CYPRUS_NATIONAL_IDENTIFICATION_NUMBER	Der zypriotische Personalausweis.
CYPRUS_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Zypern).

Datentyp	Beschreibung
CYPRUS_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Zypern).
CYPRUS_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Zypern).

Datentypen für Tschechien

Datentyp	Beschreibung
CZECHIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Tschechien).
CZECHIA_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (spezifisch für Tschechien).
CZECHIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Tschechien).

Datentypen für Dänemark

Datentyp	Beschreibung
DENMARK_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Dänemark).
DENMARK_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (spezifisch für Dänemark).
DENMARK_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Dänemark).
DENMARK_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Dänemark).

Datentypen für Estland

Datentyp	Beschreibung
ESTONIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Estland).
ESTONIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Estland).
ESTONIA_PERSONAL_IDENTIFICATION_CODE	Die persönliche Identifikationsnummer (spezifisch für Estland).
ESTONIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Estland).

Datentypen für Finnland

Datentyp	Beschreibung
FINLAND_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Finnland).
FINLAND_HEALTH_INSURANCE_NUMBER	Die Krankenversicherungsnummer (spezifisch für Finnland).
FINLAND_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für Finnland).
FINLAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Finnland).
FINLAND_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Finnland).

Datentypen für Frankreich

Datentyp	Beschreibung
FRANCE_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für Frankreich).

Datentyp	Beschreibung
FRANCE_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Frankreich).
FRANCE_HEALTH_INSURANCE_NUMBER	Nummer der französischen Krankenversicherung.
FRANCE_INSEE_CODE	Sozialversicherungs-, SSN- oder NIR-Nummer in Frankreich.
FRANCE_NATIONAL_IDENTIFICATION_NUMBER	Nationale Identifikationsnummer (CNI) für Frankreich.
FRANCE_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Frankreich).
FRANCE_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Frankreich).
FRANCE_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Frankreich).

Datentypen für Deutschland

Datentyp	Beschreibung
GERMANY_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für Deutschland).
GERMANY_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Deutschland).
GERMANY_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Deutschland).
GERMANY_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (spezifisch für Deutschland).

Datentyp	Beschreibung
GERMANY_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Deutschland).
GERMANY_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Deutschland).

Datentypen für Griechenland

Datentyp	Beschreibung
GREECE_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Griechenland).
GREECE_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Griechenland).
GREECE_SSN	Die Sozialversicherungsnummer (für griechische Personen).
GREECE_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Griechenland).
GREECE_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Griechenland).

Datentypen für Ungarn

Datentyp	Beschreibung
HUNGARY_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Ungarn).
HUNGARY_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Ungarn).
HUNGARY_SSN	Die Sozialversicherungsnummer (für ungarische Personen).

Datentyp	Beschreibung
HUNGARY_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Ungarn).
HUNGARY_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Ungarn).

Datentypen für Island

Datentyp	Beschreibung
ICELAND_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Island).
ICELAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Island).
ICELAND_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Island).

Datentypen für Indien

Datentyp	Beschreibung
INDIA_AADHAAR_NUMBER	Von der Unique Identification Authority of India ausgestellte Aadhaar-Identifikationsnummer.
INDIA_PERMANENT_ACCOUNT_NUMBER	Permanente Kontonummer (PAN) für Indien.

Datentypen für Indonesien

Datentyp	Beschreibung
INDONESIA_IDENTITY_CARD_NUMBER	Die nationale Kennung (spezifisch für Indonesien).

Datentypen für Irland

Datentyp	Beschreibung
IRELAND_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Irland).
IRELAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Irland).
IRELAND_PERSONAL_PUBLIC_SERVICE_NUMBER	Persönliche öffentliche Servicenummer (PPS) für Irland.
IRELAND_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Irland).
IRELAND_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Irland).

Datentypen für Israel

Datentyp	Beschreibung
ISRAEL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Israel).

Datentypen für Italien

Datentyp	Beschreibung
ITALY_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für Italien).
ITALY_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Italien).
ITALY_FISCAL_CODE	Die Identifikationsnummer, auch bekannt als der italienische Codice Fiscale.
ITALY_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Italien).
ITALY_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Italien).

Japanische Datentypen

Datentyp	Beschreibung
JAPAN_BANK_ACCOUNT	Japanisches Bankkonto.
JAPAN_DRIVING_LICENSE	Eine Führerscheinnummer für Japan.
JAPAN_MY_NUMBER	Die eindeutige Kennung für japanische Staatsbürger oder Unternehmen, die für Steuerverwaltung, Sozialversicherungsverwaltung und Katastrophenschutz verwendet wird
JAPAN_PASSPORT_NUMBER	Japanische Reisepassnummer.

Datentypen für Korea

Datentyp	Beschreibung
KOREA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Korea).
KOREA_RESIDENCE_REGISTRATION_NUMBER_FOR_CITIZENS	Aufenthaltsregistrierungsnummer für Einwohner Koreas.
KOREA_RESIDENCE_REGISTRATION_NUMBER_FOR_FOREIGNERS	Koreanische Aufenthaltsregistrierungsnummer für Ausländer.

Datentypen für Lettland

Datentyp	Beschreibung
LATVIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Lettland).
LATVIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Lettland).
LATVIA_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (spezifisch für Lettland).

Datentyp	Beschreibung
LATVIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Lettland).

Datentypen für Liechtenstein

Datentyp	Beschreibung
LIECHTENSTEIN_NATIONAL_IDENTIFICATIO N_NUMBER	Die nationale Kennung (spezifisch für Liechtens tein).
LIECHTENSTEIN_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Liechtens tein).
LIECHTENSTEIN_TAX_IDENTIFIC ATION_NUMBER	Steueridentifikationsnummer (spezifisch für Liechtenstein).

Datentypen für Litauen

Datentyp	Beschreibung
LITHUANIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Litauen).
LITHUANIA_PERSONAL_IDENTIFI CATION_NUMBER	Die persönliche Identifikationsnummer (spezifis ch für Litauen).
LITHUANIA_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Litauen).
LITHUANIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Litauen).

Datentypen für Luxemburg

Datentyp	Beschreibung
LUXEMBOURG_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Luxemburg).
LUXEMBURG_NATIONAL_INDIVIDUAL_NUMBER	Die nationale Kennung (spezifisch für Luxemburg).
LUXEMBOURG_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Luxemburg).
LUXEMBOURG_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Luxemburg).
LUXEMBOURG_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Luxemburg).

Datentypen in Malaysia

Datentyp	Beschreibung
MALAYSIA_MYKAD_NUMBER	Die nationale Kennung (spezifisch für Indonesien).
MALAYSIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Malaysia).

Datentypen für Malta

Datentyp	Beschreibung
MALTA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Malta).
MALTA_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Malta).
MALTA_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Malta).

Datentyp	Beschreibung
MALTA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Malta).

Datentypen für Mexiko

Datentyp	Beschreibung
MEXICO_CLABE_NUMBER	Mexiko CLABE (Clave Bancaria Estandarizada) Banknummer).
MEXICO_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Mexiko).
MEXICO_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Mexiko).
MEXICO_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Mexiko).
MEXICO_UNIQUE_POPULATION_REGISTRY_CODE	Der eindeutige Identitätscode der Clave Única de Registro de Población (CURP) für Mexiko.

Datentypen für die Niederlande

Datentyp	Beschreibung
NETHERLANDS_CITIZEN_SERVICE_NUMBER	Niederländische Staatsbürgernummer (BSN, Burgerservicenummer).
NETHERLANDS_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für die Niederlande).
NETHERLANDS_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für die Niederlande).
NETHERLANDS_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für die Niederlande).

Datentyp	Beschreibung
NETHERLANDS_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für die Niederlande).
NETHERLANDS_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für die Niederlande).

Datentypen für Neuseeland

Datentyp	Beschreibung
NEW_ZEALAND_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Neuseeland).
NEW_ZEALAND_NATIONAL_HEALTH_INDEX_NUMBER	Nummer des neuseeländischen Gesundheitsindex.
NEW_ZEALAND_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer, auch Inland Revenue Number genannt (spezifisch für Neuseeland).

Datentypen für Norwegen

Datentyp	Beschreibung
NORWAY_BIRTH_NUMBER	Norwegische nationale Identitätsnummer.
NORWAY_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Norwegen).
NORWAY_HEALTH_INSURANCE_NUMBER	Krankenversicherungsnummer für Norwegen.
NORWAY_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für Norwegen).
NORWAY_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Norwegen).

Datentypen für die Philippinen

Datentyp	Beschreibung
PHILIPPINES_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für die Philippinen).
PHILIPPINES_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für die Philippinen).

Datentypen für Polen

Datentyp	Beschreibung
POLAND_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Polen).
POLAND_IDENTIFICATION_NUMBER	Die Kennung für Polen.
POLAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Polen).
POLAND_REGON_NUMBER	Die REGON-Identifikationsnummer, auch bekannt als Statistische Identifikationsnummer.
POLAND_SSN	Die Sozialversicherungsnummer (für polnische Personen).
POLAND_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Polen).
POLAND_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Polen).

Datentypen für Portugal

Datentyp	Beschreibung
PORTUGAL_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Portugal).

Datentyp	Beschreibung
PORTUGAL_NATIONAL_IDENTIFIC ATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für Portugal).
PORTUGAL_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Portugal).
PORTUGAL_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Portugal).
PORTUGAL_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Portugal).

Datentypen für Rumänien

Datentyp	Beschreibung
ROMANIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Rumänien).
ROMANIA_NUMERICAL_PERSONAL_CODE	Die persönliche Identifikationsnummer (spezifis ch für Rumänien).
ROMANIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Rumänien).
ROMANIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Rumänien).

Datentypen für Singapur

Datentyp	Beschreibung
SINGAPORE_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Singapur).
SINGAPORE_NATIONAL_REGISTRY _IDENTIFICATION_NUMBER	Der nationale Registrierungsausweis für Singapur.

Datentyp	Beschreibung
SINGAPORE_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Singapur) .
SINGAPORE_UNIQUE_ENTITY_NUMBER	Die eindeutige Entitätsnummer für Singapur.

Datentypen für die Slowakei

Datentyp	Beschreibung
SLOVAKIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für die Slowakei).
SLOVAKIA_NATIONAL_IDENTIFIC ATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für die Slowakei).
SLOVAKIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für die Slowakei).
SLOVAKIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für die Slowakei).

Datentypen für Slowenien

Datentyp	Beschreibung
SLOVENIA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Slowenien).
SLOVENIA_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Slowenien).
SLOVENIA_TAX_IDENTIFICATION_NUMBER	Steueridentifikationsnummer (spezifisch für Slowenien).
SLOVENIA_UNIQUE_MASTER_CITI ZEN_NUMBER	Eindeutige Hauptbürgernummer (JMBG) für slowenische Staatsbürger.

Datentyp	Beschreibung
SLOVENIA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Slowenien).

Datentypen für Südafrika

Datentyp	Beschreibung
SOUTH_AFRICA_PERSONAL_IDENTIFICATION_NUMBER	Die persönliche Identifikationsnummer (spezifisch für Südafrika).

Datentypen für Spanien

Datentyp	Beschreibung
SPAIN_BANK_ACCOUNT	Die Bankkontonummer (spezifisch für Spanien).
SPANIEN_DNI	Der nationale Personalausweis (Documento Nacional de Identidad) für Spanien.
SPAIN_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch in Spanien).
SPANIEN_NIE	Die Ausländeridentitätsnummer (spezifisch für Spanien), auch bekannt als NIE.
SPAIN_NIF	Steueridentifikationsnummer (spezifisch für Spanien), auch bekannt als NIF.
SPAIN_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Spanien).
SPANIEN_SSN	Die Sozialversicherungsnummer (für Personen aus Spanien).
SPAIN_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Spanien).

Datentypen in Sri Lanka

Datentyp	Beschreibung
SRI_LANKA_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Kennung (spezifisch für Sri Lanka).

Datentypen für Schweden

Datentyp	Beschreibung
SWEDEN_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Schweden).
SWEDEN_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Schweden).
SWEDEN_PERSONAL_IDENTIFICATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für Schweden).
SWEDEN_TAX_IDENTIFICATION_NUMBER	Schwedische Steueridentifikationsnummer (Personennummer).
SWEDEN_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Schweden).

Datentypen für die Schweiz

Datentyp	Beschreibung
SWITZERLAND_AHV	Die Sozialversicherungsnummer für Schweizer Personen (AHV).
SWITZERLAND_HEALTH_INSURANCE_NUMBER	Nummer der Schweizer Krankenversicherung.
SWITZERLAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für die Schweiz).
SWITZERLAND_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für die Schweiz).

Datentypen für Thailand

Datentyp	Beschreibung
THAILAND_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für Thailand) .
THAILAND_PERSONAL_IDENTIFIC ATION_NUMBER	Die persönliche Identifikationsnummer (spezifis ch für Thailand).

Datentypen für die Türkei

Datentyp	Beschreibung
TURKEY_NATIONAL_IDENTIFICAT ION_NUMBER	Die nationale Identifikationsnummer (spezifisch für die Türkei).
TURKEY_PASSPORT_NUMBER	Die Reisepassnummer (spezifisch für die Türkei).
TURKEY_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für die Türkei).

Datentypen für die Ukraine

Datentyp	Beschreibung
UKRAINE_INDIVIDUAL_IDENTIFICATION_NU MBER	Die eindeutige Kennung (spezifisch für die Ukraine).
UKRAINE_PASSPORT_NUMBER_DOMESTIC	Die Nummer des inländischen Reisepasses (spezifisch für die Ukraine).
UKRAINE_PASSPORT_NUMBER_INT ERNATIONAL	Die Nummer des internationalen Reisepasses (spezifisch für die Ukraine).

Datentypen für die Vereinigten Arabischen Emirate (VAE)

Datentyp	Beschreibung
UNITED_ARAB_EMIRATES_PERSONAL_NUMBER	Die persönliche Identifikationsnummer (spezifisch für die VAE).

Britische Datentypen

Datentyp	Beschreibung
UK_BANK_ACCOUNT	Bankkonto im Vereinigten Königreich (UK).
UK_BANK_SORT_CODE	Bankleitzahl für das Vereinigte Königreich (UK). Bankleitzahlen sind Bankleitzahlen, die zur Weiterleitung von Geldtransfers zwischen Banken innerhalb der jeweiligen Ländern über ihre jeweiligen Clearing-Organisationen verwendet werden.
UK_DRIVING_LICENSE	Die Führerscheinnummer für das Vereinigte Königreich Großbritannien und Nordirland (spezifisch für Großbritannien)
UK_ELECTORAL_ROLL_NUMBER	Die Electoral Roll Number (ERN) ist die Identifikationsnummer, die einer Person für die Registrierung der britischen Wahlen ausgestellt wurde. Das Format dieser Nummer wird durch die britischen Regierungsstandards des britischen Kabinettsbüros festgelegt.
UK_NATIONAL_HEALTH_SERVICE_NUMBER	Die National Health Service (NHS) -Nummer ist die eindeutige Nummer, die einem registrierten Benutzer von öffentlichen Gesundheitsdiensten im Vereinigten Königreich zugewiesen wird.
UK_NATIONAL_INSURANCE_NUMBER	Die National Insurance Number (NINO) ist eine Nummer, die im Vereinigten Königreich (UK) verwendet wird, um eine Person für das

Datentyp	Beschreibung
	nationale Versicherungsprogramm oder das Sozialversicherungssystem zu identifizieren. Die Nummer wird auch als NI-Nummer oder NINO bezeichnet.
UK_PASSPORT_NUMBER	Reisepassnummer des Vereinigten Königreichs (UK).
UK_UNIQUE_TAXPAYER_REFERENCE_NUMBER	Die eindeutige Steuerzahlerreferenznummer (UTR) des Vereinigten Königreichs (UK). Eine Kennung, die von der britischen Regierung zur Verwaltung des Steuersystems verwendet wird.
UK_VALUE_ADDED_TAX	Die Mehrwertsteuer ist eine Verbrauchssteuer, die vom Endverbraucher getragen wird. Die Mehrwertsteuer wird für jede Transaktion im Herstellungs- und Vertriebsprozess gezahlt. Für das Vereinigte Königreich wird die Umsatzsteuer-Identifikationsnummer vom Umsatzsteueramt für die Region ausgestellt, in der das Unternehmen ansässig ist.
UK_PHONE_NUMBER	Telefonnummer des Vereinigten Königreichs (UK).

Datentypen für Venezuela

Datentyp	Beschreibung
VENEZUELA_DRIVING_LICENSE	Die Führerscheinnummer (spezifisch für Venezuela).
VENEZUELA_NATIONAL_IDENTIFICATION_NUMBER	Die nationale Identifikationsnummer (spezifisch für Venezuela).
VENEZUELA_VALUE_ADDED_TAX	Mehrwertsteuer (spezifisch für Venezuela).

Verwenden der detaillierten Erkennung sensibler Daten

> **Note**
>
> Feinkörnige Aktionen sind nur verfügbar in AWS Glue 3.0 und 4.0. Dies beinhaltet AWS Glue Studio Erfahrung. Änderungen am dauerhaften Auditprotokoll sind auch in Version 2.0 nicht verfügbar.
> Alle AWS Glue Studio Bei Visual-Jobs der Versionen 3.0 und 4.0 wird ein Skript erstellt, das automatisch detaillierte Aktionen verwendet. APIs

Die Transformation „Sensible Daten erkennen“ bietet die Möglichkeit, Entitäten zu erkennen, zu maskieren oder zu entfernen, die Sie definieren oder die von AWS Glue vordefiniert wurden. Mithilfe detaillierter Aktionen können Sie außerdem eine bestimmte Aktion pro Entität anwenden. Weitere Vorteile:

- Verbesserte Leistung, da Aktionen sofort angewandt werden, wenn Daten erkannt wurden.
- Sie haben die Option, bestimmte Spalten ein- oder auszuschließen.
- Sie können Datenentitäten teilweise maskieren – dadurch müssen Sie nicht die gesamte Zeichenfolge maskieren. Es werden einfache Parameter mit Offsets und Regex unterstützt.

Im Folgenden finden Sie Codefragmente zur Erkennung sensibler Daten APIs und detaillierter Aktionen, die in den Beispielaufträgen verwendet wurden, auf die im nächsten Abschnitt verwiesen wird.

Detect-API – detaillierte Aktionen verwenden den neuen Parameter `detectionParameters`:

```
def detect(
    frame: DynamicFrame,
    detectionParameters: JsonOptions,
    outputColumnName: String = "DetectedEntities",
    detectionSensitivity: String = "LOW"
): DynamicFrame = {}
```

Verwendung der Erkennung sensibler Daten mit detaillierten Aktionen APIs

Die Erkennung sensibler Daten APIs mithilfe von Detect analysiert die angegebenen Daten, bestimmt, ob es sich bei den Zeilen oder Spalten um Entitätstypen für sensible Daten handelt, und führt vom Benutzer für jeden Entitätstyp festgelegte Aktionen aus.

Verwenden der Detect-API mit detaillierten Aktionen

Verwenden Sie die Detect-API und geben Sie die Werte `outputColumnName` und `detectionParameters` an.

```
   object GlueApp {
     def main(sysArgs: Array[String]) {

       val spark: SparkContext = new SparkContext()
       val glueContext: GlueContext = new GlueContext(spark)

       // @params: [JOB_NAME]
       val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
       Job.init(args("JOB_NAME"), glueContext, args.asJava)

       // Script generated for node S3 bucket. Creates DataFrame from data stored in
S3.
       val S3bucket_node1 =
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
"\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
connectionType="s3", format="csv", options=JsonOptions("""{"paths":
["s3://189657479688-ddevansh-pii-test-bucket/tiny_pii.csv"], "recurse": true}"""),
transformationContext="S3bucket_node1").getDynamicFrame()

       // Script generated for node Detect Sensitive Data. Will run detect API for the
DataFrame
       // detectionParameter contains information on which EntityType are being
detected
       // and what actions are being applied to them when detected.
       val DetectSensitiveData_node2 = EntityDetector.detect(
           frame = S3bucket_node1,
           detectionParameters = JsonOptions(
            """
               {
                   "PHONE_NUMBER": [
                       {
```

```
                            "action": "PARTIAL_REDACT",
                            "actionOptions": {
                                "numLeftCharsToExclude": "3",
                                "numRightCharsToExclude": "4",
                                "redactChar": "#"
                            },
                            "sourceColumnsToExclude": [ "Passport No", "DL NO#" ]
                        }
                    ],
                    "USA_PASSPORT_NUMBER": [
                        {
                            "action": "SHA256_HASH",
                            "sourceColumns": [ "Passport No" ]
                        }
                    ],
                    "USA_DRIVING_LICENSE": [
                        {
                            "action": "REDACT",
                            "actionOptions": {
                                "redactText": "USA_DL"
                            },
                            "sourceColumns": [ "DL NO#" ]
                        }
                    ]

                }
            """
            ),
            outputColumnName = "DetectedEntities"
        )

        // Script generated for node S3 bucket. Store Results of detect to S3 location
        val S3bucket_node3 = glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://amzn-s3-demo-bucket/test-output/",
"partitionKeys": []}"""), transformationContext="S3bucket_node3",
format="json").writeDynamicFrame(DetectSensitiveData_node2)

        Job.commit()
    }
```

Das obige Skript erstellt ein DataFrame von einem Speicherort in Amazon S3 und führt dann die `detect` API aus. Da die `detect` API erfordert, dass das Feld `detectionParameters` (eine Zuordnung des Entitätsnamens zu einer Liste aller Aktionseinstellungen, die für diese Entität verwendet werden sollen) durch das `JsonOptions` Objekt von AWS Glue repräsentiert wird, können wir damit auch die Funktionalität der API erweitern.

Geben Sie für jede Aktion pro Entität eine Liste aller Spaltennamen ein, auf die die Kombination aus Entität und Aktion angewendet werden soll. Dadurch können Sie anpassen, welche Entitäten für jede Spalte in Ihrem Datensatz erkannt werden sollen. Entitäten, von denen Sie wissen, dass sie sich nicht in einer bestimmten Spalte befinden, können übersprungen werden. Das macht Ihre Aufträge außerdem leistungsfähiger, da keine unnötigen Erkennungsaufrufe für diese Entitäten ausgeführt werden, und ermöglicht Ihnen, individuelle Aktionen für jede Kombination aus Spalte und Entität auszuführen.

Bei näherer Betrachtung von `detectionParameters` gibt es im Beispielauftrag drei Entitätstypen. Diese sind `Phone Number`, `USA_PASSPORT_NUMBER` und `USA_DRIVING_LICENSE`. Für jeden dieser Entitätstypen führt AWS Glue unterschiedliche Aktionen aus, die entweder`PARTIAL_REDACT`, `SHA256_HASH`REDACT, und lauten`DETECT`. Im Falle einer Erkennung hat jeder der Entitätstypen außerdem `sourceColumns` für die Anwendung und/oder `sourceColumnsToExclude`.

> **Note**
>
> Pro Spalte kann nur eine edit-in-place Aktion (`PARTIAL_REDACT`SHA256_HASH,, oder`REDACT`) verwendet werden, aber die `DETECT` Aktion kann mit jeder dieser Aktionen verwendet werden.

Das Feld `detectionParameters` hat das folgende Layout:

```
ENTITY_NAME -> List[Actions]
{
 "ENTITY_NAME": [{
  Action, // required
  ColumnSpecs,
  ActionOptionsMap
    }],
    "ENTITY_NAME2": [{
  ...
    }]
```

```
    }
```

Die Typen von `actions` und `actionOptions` werden unten aufgeführt:

```
DETECT
{
    # Required
    "action": "DETECT",
    # Optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for DETECT
    },
    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

SHA256_HASH
{
    # Required
    "action": "SHA256_HASH",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for SHA256_HASH
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

REDACT
{
    # Required
```

```
    "action": "REDACT",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // The text that is being replaced
        "redactText": "USA_DL"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

PARTIAL_REDACT
{
    # Required
    "action": "PARTIAL_REDACT",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // number of characters to not redact from the left side
        "numLeftCharsToExclude": "3",
        // number of characters to not redact from the right side
        "numRightCharsToExclude": "4",
        // the partial redact will be made with this redacted character
        "redactChar": "#",
        // regex pattern for partial redaction
        "matchPattern": "[0-9]"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}
```

Wenn das Skript ausgeführt wird, werden die Ergebnisse im angegebenen Amazon-S3-Speicherort ausgegeben. Sie können Ihre Daten in Amazon S3 anzeigen, wobei die ausgewählten Entitätstypen auf der Grundlage der ausgewählten Aktion sensibilisiert werden. In dem Fall hätten wir Zeilen, die in etwa so aussehen:

```
{
    "Name": "Colby Schuster",
    "Address": "39041 Antonietta Vista, South Rodgerside, Nebraska 24151",
    "Car Owned": "Fiat",
    "Email": "Kitty46@gmail.com",
    "Company": "O'Reilly Group",
    "Job Title": "Dynamic Functionality Facilitator",
    "ITIN": "991-22-2906",
    "Username": "Cassandre.Kub43",
    "SSN": "914-22-2906",
    "DOB": "2020-08-27",
    "Phone Number": "1-2#######1718",
    "Bank Account No": "69741187",
    "Credit Card Number": "6441-6289-6867-2162-2711",
    "Passport No": "94f311e93a623c72ccb6fc46cf5f5b0265ccb42c517498a0f27fd4c43b47111e",
    "DL NO#": "USA_DL"
}
```

Im obigen Skript wurde die `Phone Number` teilweise mit # redigiert. Das `Passport No` wurde in einen SHA256 Hash geändert. Die `DL NO#` wurde als US-Führerscheinnummer erkannt und zu „USA_DL“ redigiert, wie es in `detectionParameters` angegeben war.

> **Note**
>
> Die classifyColumns-API ist aufgrund der Art der API nicht für die Verwendung mit detaillierten Aktionen verfügbar. Diese API führt spaltenbasierte Stichproben durch (vom Benutzer anpassbar, hat aber Standardwerte), um die Erkennung zu beschleunigen. Aus diesem Grund müssen bei detaillierten Aktionen alle Werte iteriert werden.

Dauerhaftes Auditprotokoll

Eine neue Funktion, die mit fein abgestuften Aktionen eingeführt wurde (aber auch verfügbar ist, wenn die normale Version verwendet wird APIs), ist das Vorhandensein eines persistenten Auditprotokolls. Derzeit wird bei der Ausführung der Detect-API ein zusätzlicher Spaltenparameter

(standardmäßig `DetectedEntities`, aber über `outputColumnName` anpassbar) mit Metadaten zur PII-Erkennung hinzugefügt. Dieser hat jetzt einen „actionUsed“ -Metadatenschlüssel, der entweder `DETECT`, `PARTIAL_REDACT`, `SHA256_HASH` oder `REDACT` lautet.

```
"DetectedEntities": {
    "Credit Card Number": [
        {
            "entityType": "CREDIT_CARD",
            "actionUsed": "DETECT",
            "start": 0,
            "end": 19
        }
    ],
    "Phone Number": [
        {
            "entityType": "PHONE_NUMBER",
            "actionUsed": "REDACT",
            "start": 0,
            "end": 14
        }
    ]
}
```

Selbst Kunden, die solche Aktionen APIs ohne detaillierte Aktionen verwenden, `detect(entityTypesToDetect, outputColumnName)` werden dieses persistente Auditprotokoll im resultierenden Datenrahmen sehen.

Kunden, die APIs mit detaillierten Aktionen arbeiten, sehen alle Aktionen, unabhängig davon, ob sie redigiert sind oder nicht. Beispiel:

```
+---------------------+---------------
+-------------------------------------------------------------------------------------------------
+
| Credit Card Number  |  Phone Number  |
                                              DetectedEntities
                                                                  |
+---------------------+---------------
+-------------------------------------------------------------------------------------------------
+
| 622126741306XXXX    | +12#####7890   | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":16}],"Phone
```

```
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":12}]}} |
| 6221 2674 1306 XXXX | +12#######7890 | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
| 6221-2674-1306-XXXX | 22#######7890  | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+
```

Wenn Sie die Spalte nicht sehen möchten, können Sie die zusätzliche DetectedEntitiesSpalte einfach in einem benutzerdefinierten Skript löschen.

AWS Glue Visual Job API

AWS Glue bietet eine API, die es Kunden ermöglicht, Datenintegrationsjobs mit dem zu erstellen AWS Glue API aus einem JSON-Objekt, das einen visuellen Schrittworkflow darstellt. Kunden können den visuellen Editor dann verwenden in AWS Glue Studio um mit diesen Jobs zu arbeiten.

Weitere Informationen zu den Datentypen der Visual Job API finden Sie unter Visual Job API.

Themen

- API-Design und CRUD APIs
- Erste Schritte
- Beschränkungen visueller Aufträge

API-Design und CRUD APIs

Die CreateJob und unterstützen UpdateJob APIs jetzt einen zusätzlichen optionalen Parameter, codeGenConfiguration Nodes. Die Angabe einer nicht leeren JSON-Struktur für dieses Feld führt dazu, dass die DAG registriert wird AWS Glue Studio für den erstellten Job und den zugehörigen Code, der generiert wird. Ein Nullwert oder eine leere Zeichenfolge für dieses Feld bei der Auftragserstellung wird ignoriert.

Aktualisierungen im Feld codeGenConfiguration Nodes erfolgen über UpdateJob AWS Glue API auf ähnliche Weise wie CreateJob. Das gesamte Feld sollte angegeben werden UpdateJob , in dem die DAG wie gewünscht geändert wurde. Ein bereitgestellter Nullwert wird ignoriert und es würde keine Aktualisierung der DAG durchgeführt. Eine leere Struktur oder Zeichenfolge führt dazu, dass die codeGenConfiguration Knoten als leer gesetzt und alle vorherigen DAG-Dateien entfernt werden. Die GetJob API gibt eine DAG zurück, falls eine existiert. Die DeleteJob API löscht auch alle zugehörigen DAG.

Erste Schritte

Verwenden Sie die CreateJob Aktion, um einen Job zu erstellen. Die `CreateJob` Anforderungseingabe wird ein zusätzliches Feld 'codeGenConfigurationKnoten' enthalten, in dem Sie das DAG-Objekt in JSON angeben können.

Beachten Sie die folgenden Dinge:

- Das Feld 'codeGenConfigurationNodes' ist eine Zuordnung von NodeID zu Knoten.
- Jeder Knoten beginnt mit einem Schlüssel, der angibt, um welche Art von Knoten es sich handelt.
- Es kann nur ein Schlüssel angegeben werden, da ein Knoten nur einen Typ haben kann.
- Das Eingabefeld enthält die übergeordneten Knoten des aktuellen Knotens.

Das Folgende ist eine JSON-Darstellung einer Eingabe. CreateJob

```
{
  "node-1": {
    "S3CatalogSource": {
      "Table": "csvFormattedTable",
      "PartitionPredicate": "",
      "Name": "S3 bucket",
      "AdditionalOptions": {},
      "Database": "myDatabase"
    }
  },
  "node-3": {
    "S3DirectTarget": {
      "Inputs": ["node-2"],
      "PartitionKeys": [],
      "Compression": "none",
      "Format": "json",
      "SchemaChangePolicy": { "EnableUpdateCatalog": false },
```

```
      "Path": "",
      "Name": "S3 bucket"
    }
  },
  "node-2": {
    "ApplyMapping": {
      "Inputs": ["node-1"],
      "Name": "ApplyMapping",
      "Mapping": [
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader1",
          "FromPath": ["myheader1"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader2",
          "FromPath": ["myheader2"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader3",
          "FromPath": ["myheader3"]
        }
      ]
    }
  }
}
```

Aktualisieren und Erhalten von Aufträgen

Da UpdateJobes auch ein Feld „codeGenConfigurationKnoten“ geben wird, wird das Eingabeformat dasselbe sein. Siehe Aktion UpdateJob.

Die GetJobAktion gibt ebenfalls ein Feld „codeGenConfigurationKnoten“ im gleichen Format zurück. Siehe Aktion GetJob.

Beschränkungen visueller Aufträge

Da der Parameter 'codeGenConfigurationNodes' dem bestehenden Parameter hinzugefügt wurde APIs, werden alle darin enthaltenen Einschränkungen APIs übernommen. Außerdem wird die Größe der codeGenConfiguration Knoten und einiger Knoten begrenzt. Siehe Auftragsstruktur für weitere Informationen.

Programmieren von Ray-Skripten

AWS Glue macht es einfach, Ray-Skripte zu schreiben und auszuführen. In diesem Abschnitt werden die unterstützten Ray-Funktionen beschrieben, die in AWS Glue für Ray verfügbar sind. Sie programmieren Ray-Skripte in Python.

Ihr benutzerdefiniertes Skript muss mit der Version von Ray kompatibel sein, die durch das `Runtime`-Feld in Ihrer Auftragsdefinition definiert ist. Weitere Informationen über `Runtime` in der Auftrags-API finden Sie unter the section called “Aufträge”. Informationen zu den einzelnen Laufzeitumgebungen finden Sie unter the section called “Unterstützte Ray-Laufzeitumgebungen”.

Themen

- Tutorial: Ein ETL-Skript AWS Glue für Ray schreiben
- Verwenden von Ray Core und Ray Data in AWS Glue for Ray
- Bereitstellen von Dateien und Python-Bibliotheken für Ray-Aufträge
- Verbindung zu Daten in Ray-Aufträgen

Tutorial: Ein ETL-Skript AWS Glue für Ray schreiben

Ray gibt Ihnen die Möglichkeit, verteilte Aufgaben nativ in Python zu schreiben und zu skalieren. AWS Glue for Ray bietet serverlose Ray-Umgebungen, auf die Sie sowohl von Jobs als auch von interaktiven Sitzungen aus zugreifen können (interaktive Ray-Sitzungen befinden sich in der Vorschauversion). Das AWS Glue Jobsystem bietet eine einheitliche Methode zur Verwaltung und Ausführung Ihrer Aufgaben — nach einem Zeitplan, über einen Trigger oder von der Konsole aus. AWS Glue

Durch die Kombination dieser AWS Glue Tools entsteht eine leistungsstarke Toolchain, die Sie zum Extrahieren, Transformieren und Laden (ETL) -Workloads verwenden können — ein beliebter Anwendungsfall für. AWS Glue In diesem Tutorial lernen Sie die Grundlagen für die Zusammenstellung dieser Lösung.

Wir unterstützen auch die Verwendung von Spark AWS Glue für Ihre ETL-Workloads. Eine Anleitung zum Schreiben eines Skripts AWS Glue für Spark finden Sie unterthe section called “Tutorial: Schreiben eines Spark-Skripts”. Weitere Informationen zu verfügbaren Engines finden Sie unter the section called “AWS Glue für Spark und AWS Glue für Ray”. Ray ist in der Lage, viele verschiedene Aufgaben in den Bereichen Analytik, Machine Learning (ML) und Anwendungsentwicklung zu bewältigen.

In diesem Tutorial extrahieren, transformieren und laden Sie einen CSV-Datensatz, der in Amazon Simple Storage Service (Amazon S3) gehostet wird. Sie beginnen mit dem Datensatz zu Reiseaufzeichnungsdaten der New York City Taxi and Limousine Commission (TLC), der in einem öffentlichen Amazon-S3-Bucket gespeichert ist. Weitere Informationen zu diesem Datensatz finden Sie im Registry der offenen Daten in AWS.

Sie transformieren Ihre Daten mit vordefinierten Transformationen, die in der Ray Data-Bibliothek verfügbar sind. Ray Data ist eine Bibliothek zur Vorbereitung von Datensätzen, die von Ray entworfen wurde und standardmäßig in AWS Glue For Ray-Umgebungen enthalten ist. Weitere Informationen zu den standardmäßig enthaltenen Bibliotheken finden Sie unter the section called “Mit Ray-Aufträgen bereitgestellte Module”. Anschließend schreiben Sie Ihre transformierten Daten in einen von Ihnen kontrollierten Amazon-S3-Bucket.

Voraussetzungen — Für dieses Tutorial benötigen Sie ein AWS Konto mit Zugriff AWS Glue auf Amazon S3.

Schritt 1: Erstellen eines Buckets in Amazon S3 zur Speicherung Ihrer Ausgabedaten

Sie benötigen einen Amazon-S3-Bucket, den Sie steuern und der als Senke für die in diesem Tutorial erstellten Daten dient. Sie können diesen Bucket mit dem folgenden Verfahren erstellen.

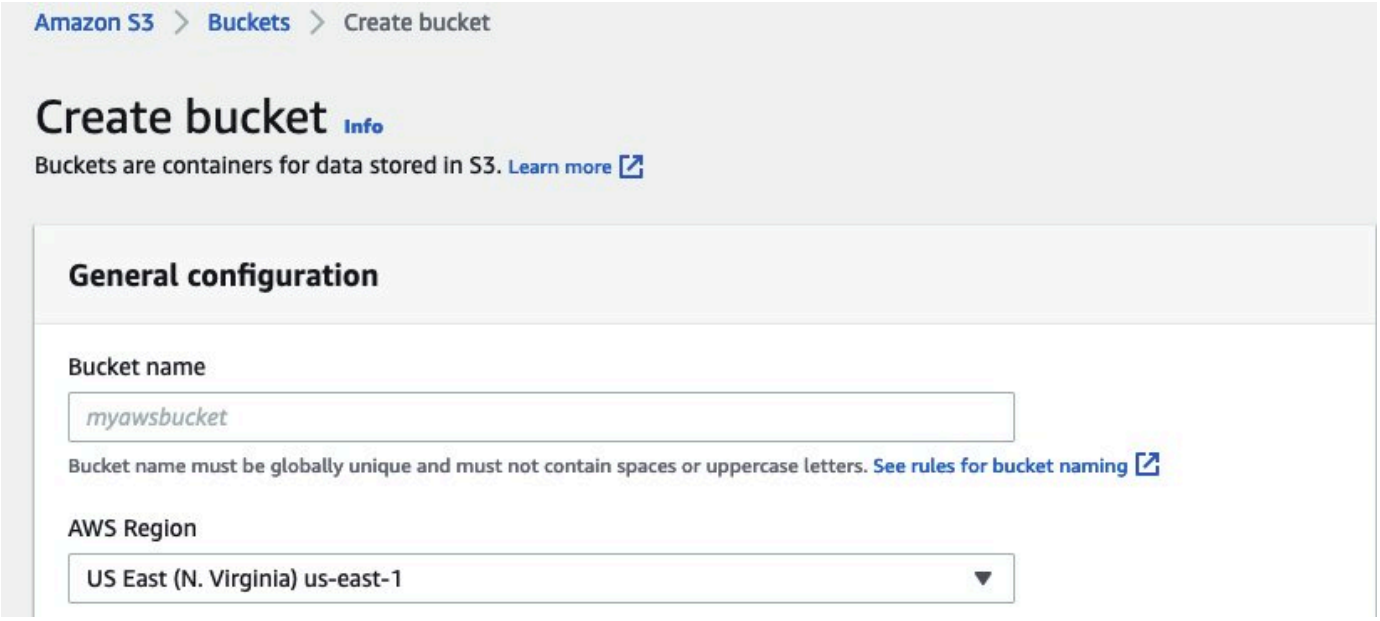
> **Note**
>
> Wenn Sie Ihre Daten in einen vorhandenen, von Ihnen kontrollierten Bucket schreiben möchten, können Sie diesen Schritt überspringen. Notieren Sie *`yourBucketName`* sich den Namen des vorhandenen Buckets, um ihn in späteren Schritten zu verwenden.

So erstellen Sie einen Bucket für die Ausgabe Ihres Ray-Auftrags

- Erstellen Sie einen Bucket, indem Sie die Schritte unter Erstellen eines Buckets im Amazon-S3-Benutzerhandbuch befolgen.

- Notieren Sie sich bei der Auswahl eines Bucket-Namens*yourBucketName*, auf den Sie sich in späteren Schritten beziehen werden.
- Für andere Konfigurationen sollten die in der Amazon-S3-Konsole vorgeschlagenen Einstellungen in diesem Tutorial gut funktionieren.

Beispielsweise könnte das Dialogfeld zur Bucket-Erstellung in der Amazon-S3-Konsole so aussehen.



Schritt 2: Erstellen einer IAM-Rolle und -Richtlinie für Ihren Ray-Auftrag

Ihr Job benötigt eine AWS Identity and Access Management (IAM-) Rolle mit folgenden Eigenschaften:

- Durch die `AWSGlueServiceRole`-verwalteten Richtlinie gewährte Berechtigungen. Dies sind die grundlegenden Berechtigungen, die für die Ausführung eines AWS Glue Jobs erforderlich sind.
- `Read`-Zugriffsebenenberechtigungen für die `nyc-tlc/*`-Amazon-S3-Ressource.
- `Write`-Zugriffsebenenberechtigungen für die *`yourBucketName`*`/*`-Amazon-S3-Ressource.
- Eine Vertrauensbeziehung, die es dem `glue.amazonaws.com`-Prinzipal ermöglicht, die Rolle zu übernehmen.

Sie können diese Rolle mit dem folgenden Verfahren erstellen.

Um eine IAM-Rolle für Ihren AWS Glue For Ray-Job zu erstellen

> **Note**
>
> Sie können eine IAM-Rolle erstellen, indem Sie viele verschiedene Verfahren befolgen. Weitere Informationen oder Optionen zum Bereitstellen von IAM-Ressourcen finden Sie in der AWS Identity and Access Management -Dokumentation.

1. Erstellen Sie eine Richtlinie, die die zuvor beschriebenen Amazon S3-Berechtigungen definiert, indem Sie die Schritte unter Erstellen von IAM-Richtlinien (Konsole) mit dem visuellen Editor im IAM-Benutzerhandbuch ausführen.
   - Wählen Sie bei der Auswahl eines Services Amazon S3.
   - Fügen Sie bei der Auswahl von Berechtigungen für Ihre Richtlinie die folgenden Aktionssätze für die folgenden Ressourcen hinzu (zuvor erwähnt):
     - Lesezugriffsebenenberechtigungen für die `nyc-tlc/*`-Amazon-S3-Ressource.
     - Schreibzugriffsebenenberechtigungen für die *`yourBucketName`*`/*`-Amazon-S3-Ressource.
   - Notieren Sie sich bei der Auswahl des Richtliniennamens*`YourPolicyName`*, auf den Sie sich in einem späteren Schritt beziehen werden.
2. Erstellen Sie eine Rolle AWS Glue für Ihren Job für Ray, indem Sie die Schritte unter Erstellen einer Rolle für einen AWS Dienst (Konsole) im IAM-Benutzerhandbuch befolgen.
   - Wenn Sie eine vertrauenswürdige AWS Serviceentität auswählen, wählen Sie`Glue`. Dadurch wird automatisch die für Ihren Auftrag erforderliche Vertrauensbeziehung ausgefüllt.
   - Fügen Sie bei der Auswahl von Richtlinien für die Berechtigungsrichtlinie die folgenden Richtlinien hinzu:
     - `AWSGlueServiceRole`
     - *`YourPolicyName`*
   - Beachten Sie bei der Auswahl des Rollennamens den Namen*`YourRoleName`*, auf den Sie sich in späteren Schritten beziehen werden.

Schritt 3: Erstellen Sie einen Job AWS Glue für Ray und führen Sie ihn aus

In diesem Schritt erstellen Sie einen AWS Glue Job mit dem AWS Management Console, stellen ihm ein Beispielskript zur Verfügung und führen den Job aus. Wenn Sie einen Auftrag erstellen, wird

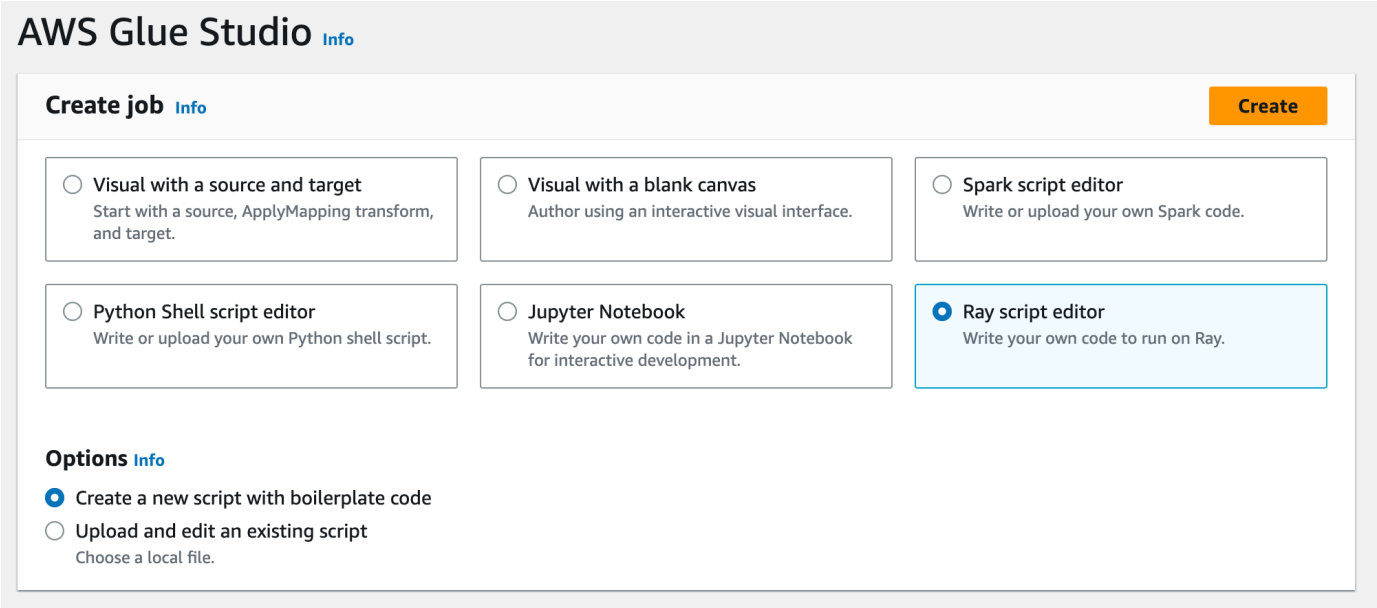
in der Konsole ein Ort angelegt, an dem Sie Ihr Ray-Skript speichern, konfigurieren und bearbeiten können. Informationen zum Erstellen von Aufgaben finden Sie unter the section called “Anmelden in der Konsole”.

In diesem Tutorial behandeln wir das folgende ETL-Szenario: Sie möchten die Datensätze vom Januar 2022 aus dem New York City TLC Trip Record-Datensatz lesen, dem Datensatz eine neue Spalte (`tip_rate`) hinzufügen, indem Sie Daten in vorhandenen Spalten kombinieren, dann eine Reihe von Spalten entfernen, die für Ihre aktuelle Analyse nicht relevant sind, und dann möchten Sie die Ergebnisse in diese schreiben*yourBucketName*. Das folgende Ray-Skript führt diese Schritte aus:

```
import ray
import pandas
from ray import data

ray.init('auto')

ds = ray.data.read_csv("s3://nyc-tlc/opendata_repo/opendata_webconvert/yellow/
yellow_tripdata_2022-01.csv")

# Add the given new column to the dataset and show the sample record after adding a new
 column
ds = ds.add_column( "tip_rate", lambda df: df["tip_amount"] / df["total_amount"])

# Dropping few columns from the underlying Dataset
ds = ds.drop_columns(["payment_type", "fare_amount", "extra", "tolls_amount",
 "improvement_surcharge"])

ds.write_parquet("s3://yourBucketName/ray/tutorial/output/")
```

Um einen For-Ray-Job AWS Glue zu erstellen und auszuführen

1. Navigieren Sie in der AWS Management Console zur AWS Glue Landingpage.
2. Wählen Sie im seitlichen Navigationsbereich ETL-Aufträge aus.
3. Wählen Sie unter Auftrag erstellen die Option Ray-Skripteditor und dann Erstellen aus, wie in der folgenden Abbildung dargestellt.



4. Fügen Sie den vollständigen Text des Skripts in den Bereich Skript ein und ersetzen Sie den vorhandenen Text.
5. Navigieren Sie zu Jobdetails und setzen Sie die Eigenschaft IAM-Rolle auf*YourRoleName*.
6. Wählen Sie Speichern und dann Ausführen aus.

Schritt 4: Überprüfen Ihrer Ausgabe

Nachdem Sie Ihren AWS Glue Job ausgeführt haben, sollten Sie überprüfen, ob die Ausgabe den Erwartungen dieses Szenarios entspricht. Sie können dies mit dem folgenden Verfahren tun.

So überprüfen Sie, ob Ihr Ray-Auftrag erfolgreich ausgeführt wurde

1. Navigieren Sie auf der Seite mit den Auftragsdetails zu Ausführungen.
2. Nach ein paar Minuten sollte eine Ausführung mit dem Ausführungsstatus Erfolgreich angezeigt werden.
3. Navigieren Sie zur Amazon S3 S3-Konsole unter https://console.aws.amazon.com/s3/und überprüfen Sie*yourBucketName*. Sie sollten Dateien finden, die in Ihren Ausgabe-Bucket geschrieben werden.
4. Lesen Sie die Parquet-Dateien und überprüfen Sie deren Inhalt. Sie können dies mit Ihren vorhandenen Tools tun. Wenn Sie nicht über ein Verfahren zur Validierung von Parquet-Dateien verfügen, können Sie dies in der AWS Glue Konsole mit einer AWS Glue interaktiven Sitzung tun, indem Sie entweder Spark oder Ray (in der Vorschauversion) verwenden.

In einer interaktiven Sitzung haben Sie Zugriff auf Ray Data-, Spark- oder Pandas-Bibliotheken, die standardmäßig bereitgestellt werden (basierend auf der von Ihnen gewählten Engine). Um den Inhalt Ihrer Datei zu überprüfen, können Sie gängige Prüfmethoden verwenden, die in diesen Bibliotheken verfügbar sind – Methoden wie `count`, `schema` und `show`. Weitere Informationen zu interaktiven Sitzungen in der Konsole finden Sie unter Notizbücher mit AWS Glue Studio verwenden und AWS Glue.

Da Sie bestätigt haben, dass Dateien in den Bucket geschrieben wurden, können Sie mit relativer Sicherheit sagen, dass Probleme bei Ihrer Ausgabe nicht mit der IAM-Konfiguration zusammenhängen. Konfigurieren Sie Ihre Sitzung mit*yourRoleName*, um Zugriff auf die entsprechenden Dateien zu haben.

Wenn Sie nicht die erwarteten Ergebnisse sehen, prüfen Sie die Fehlerbehebungsinhalte in diesem Handbuch, um die Fehlerquelle zu ermitteln und zu beheben. Den Inhalt zur Fehlerbehebung finden Sie im Fehlerbehebung AWS Glue-Kapitel. Informationen zu bestimmten Fehlern im Zusammenhang mit Ray-Aufträgen finden Sie unter the section called “Fehlerbehebung bei Ray-Fehlern” im Kapitel zur Fehlerbehebung.

Nächste Schritte

Sie haben jetzt einen ETL-Prozess gesehen und ausgeführt, der AWS Glue für Ray von Anfang bis Ende verwendet wurde. Sie können die folgenden Ressourcen verwenden, um zu verstehen, welche Tools AWS Glue für Ray zur Verfügung stehen, um Ihre Daten skalierbar zu transformieren und zu interpretieren.

- Weitere Informationen zum Aufgabenmodell von Ray finden Sie unter the section called “Verwenden von Ray Core und Ray Data in AWS Glue for Ray”. Weitere Erfahrungen mit der Verwendung von Ray-Aufgaben finden Sie in den Beispielen in der Ray-Core-Dokumentation. Lesen Sie Ray Core: Ray-Tutorials und -Beispiele (2.4.0) in der Ray-Dokumentation.
- Hinweise zu den verfügbaren Datenverwaltungsbibliotheken AWS Glue für Ray finden Sie unterthe section called “Herstellen einer Verbindung zu Daten”. Weitere Erfahrungen mit Ray Data zum Transformieren und Schreiben von Datensätzen finden Sie in den Beispielen in der Ray-Data-Dokumentation. Lesen Sie Ray Data: Beispiele (2.4.0).
- Weitere Informationen zur Konfiguration AWS Glue für Ray-Jobs finden Sie unterthe section called “Arbeiten mit Ray-Aufträgen”.

- Weitere Informationen AWS Glue zum Schreiben von Ray-Skripten finden Sie in der Dokumentation in diesem Abschnitt.

Verwenden von Ray Core und Ray Data in AWS Glue for Ray

Ray ist ein Framework zum Skalieren von Python-Skripten durch die Verteilung der Arbeit über einen Cluster. Sie können Ray als Lösung für viele Arten von Problemen verwenden. Daher bietet Ray Bibliotheken zur Optimierung bestimmter Aufgaben. In AWS Glue konzentrieren wir uns auf die Verwendung von Ray zur Transformation großer Datensätze. AWS Glue bietet Unterstützung für Ray Data und Teile von Ray Core, um diese Aufgabe zu erleichtern.

Was ist Ray Core?

Der erste Schritt beim Erstellen einer verteilten Anwendung besteht darin, Arbeiten zu identifizieren und zu definieren, die gleichzeitig ausgeführt werden können. Ray Core enthält die Teile von Ray, die Sie zum Definieren von Aufgaben verwenden, die gleichzeitig ausgeführt werden können. Ray bietet Referenz- und Schnellstartinformationen, anhand derer Sie sich mit den bereitgestellten Tools vertraut machen können. Weitere Informationen finden Sie unter Was ist Ray Core? und Ray-Core-Schnellstart. Weitere Informationen zum effektiven Definieren gleichzeitiger Aufgaben in Ray finden Sie unter Tipps für Erstanwender.

> **Aufgaben und Akteure von Ray**
>
> In der AWS Glue Ray-Dokumentation könnten wir uns auf Aufgaben und Akteure beziehen, die Kernkonzepte von Ray sind.
> Ray verwendet Python-Funktionen und -Klassen als Bausteine eines verteilten Computersystems. Ähnlich wie Python-Funktionen und -Variablen zu „Methoden“ und „Attributen“ werden, wenn sie in einer Klasse verwendet werden, werden Funktionen zu „Aufgaben“ und Klassen zu „Akteuren“, wenn sie in Ray zum Senden von Code an Worker verwendet werden. Sie können Funktionen und Klassen, die von Ray verwendet werden könnten, anhand der `@ray.remote`-Anmerkung identifizieren.
> Aufgaben und Akteure sind konfigurierbar, sie haben einen Lebenszyklus und beanspruchen während ihres gesamten Lebens Rechenressourcen. Code, der Fehler auslöst, kann auf eine Aufgabe oder einen Akteur zurückgeführt werden, wenn Sie die Grundursache von Problemen ermitteln. Daher könnten diese Begriffe auftauchen, wenn Sie lernen, wie man Ray-Jobs konfiguriert, überwacht oder AWS Glue debuggt.

Informationen zum effektiven Einsatz von Aufgaben und Akteuren zum Erstellen einer verteilten Anwendung finden Sie unter Schlüsselkonzepte in den Ray-Dokumenten.

Ray Core ist AWS Glue für Ray

AWS Glue for Ray-Umgebungen verwalten die Clusterbildung und Skalierung sowie das Sammeln und Visualisieren von Protokollen. Da wir uns um diese Probleme kümmern, beschränken wir daher den Zugriff auf und den Support für den APIs internen Ray Core, der zur Lösung dieser Probleme in einem Open-Source-Cluster verwendet würde.

In der verwalteten `Ray2.4`-Laufzeitumgebung unterstützen wir Folgendes nicht:

- Ray-Core-CLI
- Ray-State-CLI
- Metrische Hilfsmethoden von `ray.util.metrics` Prometheus:
  - Zähler
  - Messinstrument
  - Histogramm
- Weitere Debugging-Tools:
  - ray.util.pdb.set_trace
  - ray.util.inspect_serializability
  - ray.timeline

Was ist Ray Data?

Wenn Sie eine Verbindung zu Datenquellen und -zielen herstellen, Datensätze verarbeiten und allgemeine Transformationen initiieren, ist Ray Data eine unkomplizierte Methode für den Einsatz von Ray zur Lösung von Problemen bei der Transformation von Ray-Datensätzen. Weitere Informationen zur Verwendung von Ray Datafinden Sie unter Ray-Datensätze: Verteilte Datenvorverarbeitung.

Sie können Ray Data oder andere Tools verwenden, um auf Ihre Daten zuzugreifen. Weitere Informationen zum Zugriff auf Ihre Daten in Ray finden Sie unter the section called “Herstellen einer Verbindung zu Daten”.

Ray Data ist AWS Glue für Ray

Ray Data wird standardmäßig in der verwalteten `Ray2.4`-Laufzeitumgebung unterstützt und bereitgestellt. Weitere Informationen zu bereitgestellten Modulen finden Sie unter the section called “Mit Ray-Aufträgen bereitgestellte Module”.

Bereitstellen von Dateien und Python-Bibliotheken für Ray-Aufträge

Dieser Abschnitt enthält Informationen, die Sie für die Verwendung von Python-Bibliotheken mit AWS Glue Ray-Jobs benötigen. Sie können bestimmte allgemeine Bibliotheken verwenden, die standardmäßig in allen Ray-Aufträgen enthalten sind. Sie können auch Ihre eigenen Python-Bibliotheken für Ihren Ray-Auftrag bereitstellen.

Mit Ray-Aufträgen bereitgestellte Module

Sie können Datenintegrations-Workflows in einem Ray-Auftrag mit den folgenden bereitgestellten Paketen ausführen. Diese Pakete sind standardmäßig in Ray-Aufträgen verfügbar.

AWS Glue version 4.0

In AWS Glue 4.0 bietet die Ray-Umgebung (`Ray2.4`Laufzeit) die folgenden Pakete:

- boto3 == 1.26.133
- ray == 2.4.0
- pyarrow == 11.0.0
- pandas == 1.5.3
- numpy == 1.24.3
- fsspec == 2023.4.0

Diese Liste enthält alle Pakete, die mit `ray[data] == 2.4.0` installiert werden würden. Ray Data wird standardmäßig unterstützt.

Bereitstellen von Dateien für Ihren Ray-Auftrag

Sie können Ihrem Ray-Auftrag Dateien mit dem `--working-dir`-Parameter bereitstellen. Geben Sie für diesen Parameter einen Pfad zu einer ZIP-Datei an, die auf Amazon S3 gehostet wird. Innerhalb der ZIP-Datei müssen Ihre Dateien in einem einzigen Verzeichnis der obersten Ebene enthalten sein. Auf der obersten Ebene sollten sich keine anderen Dateien befinden.

Ihre Dateien werden an jeden Ray-Knoten verteilt, bevor Ihr Skript ausgeführt wird. Überlegen Sie, welche Auswirkungen dies auf den Speicherplatz haben könnte, der jedem Ray-Knoten zur Verfügung steht. Der verfügbare Festplattenspeicher wird durch den in der Auftragskonfiguration WorkerType festgelegten Wert bestimmt. Wenn Sie Ihre Auftragsdaten in großem Umfang bereitstellen möchten, ist diese Methode nicht die richtige Lösung. Weitere Informationen zum Bereitstellen von Daten für Ihren Auftrag finden Sie unter the section called “Herstellen einer Verbindung zu Daten”.

Der Zugriff auf Ihre Dateien erfolgt, wenn das Verzeichnis Ray durch den `working_dir`-Parameter bereitgestellt wird. Um beispielsweise eine Datei mit dem Namen `sample.txt` im Verzeichnis der obersten Ebene Ihrer ZIP-Datei zu lesen, können Sie Folgendes aufrufen:

```
@ray.remote
def do_work():
    f = open("sample.txt", "r")
    print(f.read())
```

Weitere Informationen zu `working_dir` finden Sie in der Ray-Dokumentation. Dieses Feature verhält sich ähnlich wie die nativen Funktionen von Ray.

Zusätzliche Python-Module für Ray-Aufträge

Zusätzliche Module aus PyPI

Ray-Aufträge verwenden den Python Package Installer (pip3), um zusätzliche Module zur Verwendung mit einem Ray-Skript zu installieren. Sie können den Parameter „`--pip-install`“ mit verschiedenen kommagetrennten Python-Modulen verwenden, um ein neues Modul hinzuzufügen oder die Version eines vorhandenen Moduls zu ändern.

Um beispielsweise ein `scikit-learn`-Modul zu aktualisieren oder ein neues hinzuzufügen, verwenden Sie das folgende Schlüssel-Wert-Paar:

```
"--pip-install", "scikit-learn==0.21.3"
```

Wenn Sie über benutzerdefinierte Module oder benutzerdefinierte Patches verfügen, können Sie mit dem `--s3-py-modules`-Parameter Ihre eigenen Bibliotheken aus Amazon S3 verteilen. Vor dem Hochladen Ihrer Distribution muss diese möglicherweise neu gepackt und erstellt werden. Befolgen Sie den unter the section called “Einbindung von Python-Code in Ray-Aufträge” beschriebenen Richtlinien.

Benutzerdefinierte Distributionen von Amazon S3

Benutzerdefinierte Verteilungen sollten sich an die Ray-Paketrichtlinien für Abhängigkeiten halten. Im folgenden Abschnitt erfahren Sie, wie Sie diese Verteilungen erstellen können. Weitere Informationen darüber, wie Ray Abhängigkeiten einrichtet, finden Sie unter Umgebungsabhängigkeiten in der Ray-Dokumentation.

Um eine benutzerdefinierte Datei nach Prüfung ihres Inhalts einzubinden, laden Sie Ihre Datei in einen Bucket hoch, der für die IAM-Rolle des Auftrags verfügbar ist. Geben Sie in Ihrer Parameterkonfiguration den Amazon-S3-Pfad zu einem Python-Zip-Archiv an. Wenn Sie mehrere verteilbare Dateien bereitstellen, trennen Sie diese durch Kommas. Zum Beispiel:

```
"--s3-py-modules", "s3://s3bucket/pythonPackage.zip"
```

Einschränkungen

Ray-Aufträge unterstützen die Kompilierung von nativem Code in der Auftragsumgebung nicht. Sie können dadurch eingeschränkt werden, wenn Ihre Python-Abhängigkeiten transitiv von nativem, kompiliertem Code abhängen. Ray-Jobs können bereitgestellte Binärdateien ausführen, sie müssen jedoch für Linux kompiliert werden. ARM64 Dies bedeutet, dass Sie möglicherweise den Inhalt von `aarch64manylinux`-Rädern verwenden können. Sie können Ihre nativen Abhängigkeiten in kompilierter Form bereitstellen, indem Sie ein Rad nach Ray-Standards neu verpacken. In der Regel bedeutet dies, dass Sie `dist-info`-Ordner entfernen, so dass sich nur noch ein Ordner im Stammverzeichnis des Archivs befindet.

Sie können die Version von `ray` oder `ray[data]` mit diesem Parameter nicht aktualisieren. Um eine neue Version von Ray zu verwenden, müssen Sie das Laufzeitfeld in Ihrem Auftrag ändern, nachdem wir die Unterstützung dafür veröffentlicht haben. Weitere Informationen zu unterstützten Ray-Versionen finden Sie unter the section called “AWS Glue Versionen”.

Einbindung von Python-Code in Ray-Aufträge

Die Python Software Foundation bietet standardisierte Verhaltensweisen für die Paketierung von Python-Dateien zur Verwendung in verschiedenen Laufzeiten. Ray führt Einschränkungen bei den Paketierungsstandards ein, die Sie beachten sollten. AWS Glue legt keine anderen Paketierungsstandards fest als die, die für Ray angegeben sind. Die folgenden Anweisungen bieten eine Standardanleitung zum Verpacken einfacher Python-Pakete.

Verpacken Sie Ihre Dateien in einem `.zip`-Archiv. Im Stammverzeichnis des Archivs sollte sich ein Verzeichnis befinden. Auf der Stammebene des Archivs sollten sich keine anderen Dateien befinden, da dies sonst zu unerwartetem Verhalten führen kann. Das Stammverzeichnis ist das Paket, und sein Name wird verwendet, um beim Importieren auf Ihren Python-Code zu verweisen.

Wenn Sie einem Ray-Auftrag mit `--s3-py-modules` eine Verteilung in dieser Form bereitstellen, können Sie Python-Code aus Ihrem Paket in Ihr Ray-Skript importieren.

Ihr Paket kann ein einzelnes Python-Modul mit einigen Python-Dateien bereitstellen, oder Sie können mehrere Module bündeln. Wenn Sie Abhängigkeiten, wie z. B. Bibliotheken von PyPI, neu bündeln, überprüfen Sie, ob in diesen Paketen versteckte Dateien und Metadatenverzeichnisse enthalten sind.

> **Warning**
>
> Bestimmte Verhaltensweisen des Betriebssystems machen es schwierig, diese Verpackungsanweisungen ordnungsgemäß zu befolgen.
>
> - OSX fügt Ihrer ZIP-Datei möglicherweise auf der obersten Ebene versteckte Dateien wie `__MACOSX` hinzu.
> - Windows fügt Ihre Dateien möglicherweise automatisch einem Ordner innerhalb der ZIP-Datei hinzu und erstellt so unbeabsichtigt einen verschachtelten Ordner.

Bei den folgenden Verfahren wird davon ausgegangen, dass Sie mit Ihren Dateien in Amazon Linux 2 oder einem ähnlichen Betriebssystem interagieren, das eine Verteilung der Info-ZIP-Hilfsprogramme `zip` und `zipinfo` bereitstellt. Wir empfehlen die Verwendung dieser Tools, um unerwartetes Verhalten zu verhindern.

So bündeln Sie Python-Dateien für die Nutzung in Ray

1. Erstellen Sie ein temporäres Verzeichnis mit Ihrem Paketnamen und bestätigen Sie dann, dass es sich bei Ihrem Arbeitsverzeichnis um das übergeordnete Verzeichnis handelt. Sie können dies mit den folgenden Befehlen tun:

   ```
   cd parent_directory
   mkdir temp_dir
   ```

2. Kopieren Sie Ihre Dateien in das temporäre Verzeichnis und bestätigen Sie dann Ihre Verzeichnisstruktur. Auf den Inhalt dieses Verzeichnisses wird direkt als Ihr Python-Modul zugegriffen. Führen Sie dazu den folgenden -Befehl aus:

   ```
   ls -AR temp_dir
   # my_file_1.py
   # my_file_2.py
   ```

3. Komprimieren Sie Ihren temporären Ordner mit ZIP. Sie können dies mit den folgenden Befehlen tun:

```
zip -r zip_file.zip temp_dir
```

4. Bestätigen Sie, dass Ihre Datei ordnungsgemäß verpackt ist. *zip_file*.zip sollte sich nun in Ihrem Arbeitsverzeichnis befinden. Sie können es mit dem folgenden Befehl überprüfen:

```
zipinfo -1 zip_file.zip
# temp_dir/
# temp_dir/my_file_1.py
# temp_dir/my_file_2.py
```

So verpacken Sie ein Python-Paket zur Nutzung in Ray neu.

1. Erstellen Sie ein temporäres Verzeichnis mit Ihrem Paketnamen und bestätigen Sie dann, dass es sich bei Ihrem Arbeitsverzeichnis um das übergeordnete Verzeichnis handelt. Sie können dies mit den folgenden Befehlen tun:

```
cd parent_directory
mkdir temp_dir
```

2. Dekomprimieren Sie Ihr Paket und kopieren Sie den Inhalt in Ihr temporäres Verzeichnis. Entfernen Sie Dateien, die sich auf Ihren vorherigen Verpackungsstandard beziehen, sodass nur der Inhalt des Moduls übrig bleibt. Bestätigen Sie mit dem folgenden Befehl, dass Ihre Dateistruktur korrekt aussieht:

```
ls -AR temp_dir
# my_module
# my_module/__init__.py
# my_module/my_file_1.py
# my_module/my_submodule/__init__.py
# my_module/my_submodule/my_file_2.py
# my_module/my_submodule/my_file_3.py
```

3. Komprimieren Sie Ihren temporären Ordner mit ZIP. Sie können dies mit den folgenden Befehlen tun:

```
zip -r zip_file.zip temp_dir
```

4. Bestätigen Sie, dass Ihre Datei ordnungsgemäß verpackt ist. *zip_file*.zip sollte sich nun in Ihrem Arbeitsverzeichnis befinden. Sie können es mit dem folgenden Befehl überprüfen:

```
zipinfo -1 zip_file.zip
# temp_dir/my_module/
# temp_dir/my_module/__init__.py
# temp_dir/my_module/my_file_1.py
# temp_dir/my_module/my_submodule/
# temp_dir/my_module/my_submodule/__init__.py
# temp_dir/my_module/my_submodule/my_file_2.py
# temp_dir/my_module/my_submodule/my_file_3.py
```

Verbindung zu Daten in Ray-Aufträgen

AWS Glue Ray-Jobs können eine breite Palette von Python-Paketen verwenden, mit denen Sie Daten schnell integrieren können. Wir stellen einen minimalen Satz an Abhängigkeiten bereit, um Ihre Umgebung nicht zu überladen. Weitere Informationen darüber, was standardmäßig enthalten ist, finden Sie unter the section called “Mit Ray-Aufträgen bereitgestellte Module”.

> **Note**
>
> AWS Glue Extrahieren, Transformieren und Laden (ETL) bietet die DynamicFrame Abstraktion zur Optimierung von ETL-Workflows, bei denen Sie Schemaunterschiede zwischen Zeilen in Ihrem Datensatz beheben. AWS Glue ETL bietet zusätzliche Funktionen — Job-Lesezeichen und Gruppierung von Eingabedateien. Wir bieten derzeit keine entsprechenden Features in Ray-Aufträgen an.
> AWS Glue for Spark bietet direkte Unterstützung für die Verbindung zu bestimmten Datenformaten, Quellen und Datenquellen. In Ray decken das AWS -SDK für Pandas und aktuelle Bibliotheken von Drittanbietern diesen Bedarf weitgehend ab. Sie müssen diese Bibliotheken zu Rate ziehen, um zu verstehen, welche Funktionen verfügbar sind.

AWS Glue für Ray ist die Integration mit Amazon VPC derzeit nicht verfügbar. Auf Ressourcen in Amazon VPC kann ohne eine öffentliche Route nicht zugegriffen werden. Weitere Informationen zur Verwendung AWS Glue mit Amazon VPC finden Sie unterthe section called “Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS PrivateLink für AWS Glue ”.

Allgemeine Bibliotheken für die Arbeit mit Daten in Ray

Ray Data – Ray Data bietet Methoden zum Umgang mit gängigen Datenformaten, Quellen und Senken. Weitere Informationen zu unterstützten Formaten und Quellen in Ray Data finden Sie unter Eingabe/Ausgabe in der Ray Data-Dokumentation. Bei Ray Data handelt es sich eher um eine eigensinnige Bibliothek als um eine Allzweckbibliothek für die Verarbeitung von Datensätzen.

Ray bietet bestimmte Hinweise zu Anwendungsfällen, in denen Ray Data möglicherweise die beste Lösung für Ihre Aufgabe sein könnte. Weitere Informationen finden Sie unter Ray-Anwendungsfälle in der Ray-Dokumentation.

AWS SDK für Pandas (awswrangler) — AWS SDK für Pandas ist ein AWS Produkt, das saubere, getestete Lösungen für das Lesen und Schreiben von AWS Diensten bietet, wenn Ihre Transformationen Daten mit Pandas verwalten. DataFrames Weitere Informationen zu den unterstützten Formaten und Quellen im SDK für Pandas finden Sie in der API-Referenz in der Dokumentation zum AWS SDK für Pandas. AWS

Beispiele zum Lesen und Schreiben von Daten mit dem AWS SDK für Pandas finden Sie unter Schnellstart in der Dokumentation zum AWS SDK für Pandas. Das AWS SDK für Pandas bietet keine Transformationen für Ihre Daten. Es bietet lediglich Unterstützung für das Lesen und Schreiben aus Quellen.

Modin – Modin ist eine Python-Bibliothek, die gängige Pandas-Operationen auf verteilbare Weise implementiert. Weitere Informationen zu Modin finden Sie in der Modin-Dokumentation. Modin selbst bietet keine Unterstützung für das Lesen und Schreiben aus Quellen. Es bietet verteilte Implementierungen gängiger Transformationen. Modin wird vom AWS SDK für Pandas unterstützt.

Wenn Sie Modin und das AWS SDK für Pandas zusammen in einer Ray-Umgebung ausführen, können Sie allgemeine ETL-Aufgaben mit leistungsstarken Ergebnissen ausführen. Weitere Informationen zur Verwendung von Modin mit dem AWS SDK für Pandas finden Sie unter Maßstabsgetreu in der AWS Dokumentation zum SDK für Pandas.

Andere Frameworks — Weitere Informationen zu Frameworks, die Ray unterstützt, finden Sie in der Ray-Dokumentation unter The Ray Ecosystem. Wir bieten keine Unterstützung für andere Frameworks in AWS Glue for Ray.

Herstellen einer Verbindung zu Daten über den Data Catalog

Die Verwaltung Ihrer Daten über den Datenkatalog in Verbindung mit Ray-Jobs wird mit dem AWS SDK für Pandas unterstützt. Weitere Informationen finden Sie im Glue-Katalog auf der AWS SDK for Pandas-Website.

Verwenden dieses Dienstes mit einem AWS SDK

AWS Software Development Kits (SDKs) sind für viele gängige Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Codebeispiele
AWS SDK für Go	AWS SDK für Go Codebeispiele
AWS SDK für Java	AWS SDK für Java Codebeispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Codebeispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Codebeispiele
AWS SDK für .NET	AWS SDK für .NET Codebeispiele
AWS SDK für PHP	AWS SDK für PHP Codebeispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Codebeispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Codebeispiele
AWS SDK für Ruby	AWS SDK für Ruby Codebeispiele
AWS SDK for Rust	AWS SDK for Rust Codebeispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Codebeispiele
AWS SDK for Swift	AWS SDK for Swift Codebeispiele

Weitere Beispiele speziell für diesen Service finden Sie unter AWS Glue API-Codebeispiele mit AWS SDKs.

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

AWS Glue API

In diesem Abschnitt werden Datentypen und Primitive beschrieben, die von AWS Glue SDKs und Tools verwendet werden. Es gibt drei allgemeine Möglichkeiten zur AWS Glue programmgesteuerten Interaktion außerhalb von AWS Management Console, für jede davon gibt es eine eigene Dokumentation:

- Sprach-SDK-Bibliotheken ermöglichen Ihnen den Zugriff auf AWS -Ressourcen aus gängigen Programmiersprachen. Weitere Informationen finden Sie unter Tools to Build on AWS (Tools zum Bauen auf)..
- Das AWS CLI ermöglicht Ihnen den Zugriff auf AWS Ressourcen von der Befehlszeile aus. Weitere Informationen finden Sie unter AWS CLI Command Reference (-Befehlsreferenz).
- AWS CloudFormation ermöglicht es Ihnen, eine Reihe von AWS Ressourcen zu definieren, die konsistent zusammen bereitgestellt werden sollen. Weitere Informationen finden Sie unter AWS CloudFormation Referenz zum AWS Glue Ressourcentyp.

In diesem Abschnitt werden gemeinsam genutzte Primitive unabhängig von diesen SDKs und Tools dokumentiert. Tools verwenden die AWS Glue Web-API-Referenz, um mit AWS ihnen zu kommunizieren.

Inhalt

- Sicherheit APIs in AWS Glue
  - Datentypen
  - DataCatalogEncryptionSettings Struktur
  - EncryptionAtRest Struktur
  - ConnectionPasswordEncryption Struktur
  - EncryptionConfiguration Struktur
  - S3Encryption-Struktur
  - CloudWatchEncryption Struktur
  - JobBookmarksEncryption Struktur
  - SecurityConfiguration Struktur
  - GluePolicy Struktur
  - DataQualityEncryption Struktur

- Operationen
- GetDataCatalogEncryptionSettings Aktion (Python: get_data_catalog_encryption_settings)
- PutDataCatalogEncryptionSettings Aktion (Python: put_data_catalog_encryption_settings)
- PutResourcePolicy Aktion (Python: put_resource_policy)
- GetResourcePolicy Aktion (Python: get_resource_policy)
- DeleteResourcePolicy Aktion (Python: delete_resource_policy)
- CreateSecurityConfiguration Aktion (Python: create_security_configuration)
- DeleteSecurityConfiguration Aktion (Python: delete_security_configuration)
- GetSecurityConfiguration Aktion (Python: get_security_configuration)
- GetSecurityConfigurations Aktion (Python: get_security_configurations)
- GetResourcePolicies Aktion (Python: get_resource_policies)
- API für Katalogobjekte
  - Kataloge-API
    - Datentypen
    - Struktur des Katalogs
    - CatalogInput Struktur
    - TargetRedshiftCatalog Struktur
    - CatalogProperties Struktur
    - CatalogPropertiesOutput Struktur
    - DataLakeAccessProperties Struktur
    - DataLakeAccessPropertiesOutput Struktur
    - FederatedCatalog Struktur
    - Operationen
    - CreateCatalog Aktion (Python: create_catalog)
    - UpdateCatalog Aktion (Python: update_catalog)
    - DeleteCatalog Aktion (Python: delete_catalog)
    - GetCatalog Aktion (Python: get_catalog)
    - GetCatalogs Aktion (Python: get_catalogs)
  - Datenbank-API
    - Datentypen

    - Database-Struktur
    - DatabaseInput Struktur
    - PrincipalPermissions Struktur
    - DataLakePrincipal Struktur
    - DatabaseIdentifier Struktur
    - FederatedDatabase Struktur
    - Operationen
    - CreateDatabase Aktion (Python: create_database)
    - UpdateDatabase Aktion (Python: update_database)
    - DeleteDatabase Aktion (Python: delete_database)
    - GetDatabase Aktion (Python: get_database)
    - GetDatabases Aktion (Python: get_databases)
- Tabellen-API
    - Datentypen
    - Tabellenstruktur
    - TableInput Struktur
    - FederatedTable Struktur
    - Spaltenstruktur
    - StorageDescriptor Struktur
    - SchemaReference Struktur
    - SerDeInfo Struktur
    - Order-Struktur
    - SkewedInfo Struktur
    - TableVersion Struktur
    - TableError Struktur
    - TableVersionError Struktur
    - SortCriterion Struktur
    - TableIdentifier Struktur
    - KeySchemaElement Struktur
    - PartitionIndex Struktur

- PartitionIndexDescriptor Struktur
- BackfillError Struktur
- IcebergInput Struktur
- OpenTableFormatInput Struktur
- ViewDefinition Struktur
- ViewDefinitionInput Struktur
- ViewRepresentation Struktur
- ViewRepresentationInput Struktur
- UpdateOpenTableFormatInput Struktur
- UpdateIcebergInput Struktur
- CreateIcebergTableInput Struktur
- UpdateIcebergTableInput Struktur
- IcebergSortOrder Struktur
- IcebergSortField Struktur
- IcebergPartitionSpec Struktur
- IcebergPartitionField Struktur
- IcebergSchema Struktur
- IcebergStructField Struktur
- IcebergTableUpdate Struktur
- Operationen
- CreateTable Aktion (Python: create_table)
- UpdateTable Aktion (Python: update_table)
- DeleteTable Aktion (Python: delete_table)
- BatchDeleteTable Aktion (Python: batch_delete_table)
- GetTable Aktion (Python: get_table)
- GetTables Aktion (Python: get_tables)
- GetTableVersion Aktion (Python: get_table_version)
- GetTableVersions Aktion (Python: get_table_versions)
- DeleteTableVersion Aktion (Python: delete_table_version)
- BatchDeleteTableVersion Aktion (Python: batch_delete_table_version)

- SearchTables Aktion (Python: search_tables)
- GetPartitionIndexes Aktion (Python: get_partition_indexes)
- CreatePartitionIndex Aktion (Python: create_partition_index)
- DeletePartitionIndex Aktion (Python: delete_partition_index)
- GetColumnStatisticsForTable Aktion (Python: get_column_statistics_for_table)
- UpdateColumnStatisticsForTable Aktion (Python: update_column_statistics_for_table)
- DeleteColumnStatisticsForTable Aktion (Python: delete_column_statistics_for_table)

- Partitions-API
  - Datentypen
  - Partitionsstruktur
  - PartitionInput Struktur
  - PartitionSpecWithSharedStorageDescriptor Struktur
  - PartitionListComposingSpec Struktur
  - PartitionSpecProxy Struktur
  - PartitionValueList Struktur
  - Segmentstruktur
  - PartitionError Struktur
  - BatchUpdatePartitionFailureEntry Struktur
  - BatchUpdatePartitionRequestEntry Struktur
  - StorageDescriptor Struktur
  - SchemaReference Struktur
  - SerDeInfo Struktur
  - SkewedInfo Struktur
  - Operationen
  - CreatePartition Aktion (Python: create_partition)
  - BatchCreatePartition Aktion (Python: batch_create_partition)
  - UpdatePartition Aktion (Python: update_partition)
  - DeletePartition Aktion (Python: delete_partition)
  - BatchDeletePartition Aktion (Python: batch_delete_partition)
  - GetPartition Aktion (Python: get_partition)

- GetPartitions Aktion (Python: get_partitions)
- BatchGetPartition Aktion (Python: batch_get_partition)
- BatchUpdatePartition Aktion (Python: batch_update_partition)
- GetColumnStatisticsForPartition Aktion (Python: get_column_statistics_for_partition)
- UpdateColumnStatisticsForPartition Aktion (Python: update_column_statistics_for_partition)
- DeleteColumnStatisticsForPartition Aktion (Python: delete_column_statistics_for_partition)

- Verbindungs-API
  - Verbindungs-API
    - Datentypen
    - Connection-Struktur
    - ConnectionInput Struktur
    - TestConnectionInput Struktur
    - PhysicalConnectionRequirements Struktur
    - GetConnectionsFilter Struktur
    - AuthenticationConfiguration Struktur
    - AuthenticationConfigurationInput Struktur
    - OAuth2Struktur der Eigenschaften
    - OAuth2PropertiesInput Struktur
    - OAuth2ClientApplication Struktur
    - AuthorizationCodeProperties Struktur
    - BasicAuthenticationCredentials Struktur
    - OAuth2Struktur der Anmeldeinformationen
    - Operationen
    - CreateConnection Aktion (Python: create_connection)
    - DeleteConnection Aktion (Python: delete_connection)
    - GetConnection Aktion (Python: get_connection)
    - GetConnections Aktion (Python: get_connections)
    - UpdateConnection Aktion (Python: update_connection)
    - TestConnection Aktion (Python: test_connection)
    - BatchDeleteConnection Aktion (Python: batch_delete_connection)

- API für Verbindungstypen
  - Verwaltung der Verbindungen APIs
  - DescribeConnectionType Aktion (Python: describe_connection_type)
  - ListConnectionTypes Aktion (Python: list_connection_types)
  - ConnectionTypeBrief Struktur
  - ConnectionTypeVariant Struktur
  - Datentypen
  - Struktur der Validierung
  - AuthConfiguration Struktur
  - Struktur der Fähigkeiten
  - Struktur der Immobilie
  - AllowedValue Struktur
  - ComputeEnvironmentConfiguration Struktur
- Verbindungsmetadaten und Vorschau-API
  - Datentypen
  - Struktur der Entität
  - Struktur des Feldes
  - Operationen
  - ListEntities Aktion (Python: list_entities)
  - DescribeEntity Aktion (Python: describe_entity)
  - GetEntityRecords Aktion (Python: get_entity_records)
- Benutzerdefinierte Funktion (API)
  - Datentypen
  - UserDefinedFunction Struktur
  - UserDefinedFunctionInput Struktur
  - Operationen
  - CreateUserDefinedFunction Aktion (Python: create_user_defined_function)
  - UpdateUserDefinedFunction Aktion (Python: update_user_defined_function)
  - DeleteUserDefinedFunction Aktion (Python: delete_user_defined_function)
  - GetUserDefinedFunction Aktion (Python: get_user_defined_function)

        - GetUserDefinedFunctions Aktion (Python: get_user_defined_functions)
    - Einen Athena Katalog importieren nach AWS Glue
        - Datentypen
        - CatalogImportStatus Struktur
        - Operationen
        - ImportCatalogToGlue Aktion (Python: import_catalog_to_glue)
        - GetCatalogImportStatus Aktion (Python: get_catalog_import_status)
- Tabellenoptimierer-API
    - Datentypen
    - TableOptimizer Struktur
    - TableOptimizerConfiguration Struktur
    - TableOptimizerVpcConfiguration Struktur
    - CompactionConfiguration Struktur
    - IcebergCompactionConfiguration Struktur
    - TableOptimizerRun Struktur
    - BatchGetTableOptimizerEntry Struktur
    - BatchTableOptimizer Struktur
    - BatchGetTableOptimizerError Struktur
    - RetentionConfiguration Struktur
    - IcebergRetentionConfiguration Struktur
    - OrphanFileDeletionConfiguration Struktur
    - IcebergOrphanFileDeletionConfiguration Struktur
    - CompactionMetrics Struktur
    - RetentionMetrics Struktur
    - OrphanFileDeletionMetrics Struktur
    - IcebergCompactionMetrics Struktur
    - IcebergRetentionMetrics Struktur
    - IcebergOrphanFileDeletionMetrics Struktur
    - RunMetrics Struktur
    - Operationen

- GetTableOptimizer Aktion (Python: get_table_optimizer)
- BatchGetTableOptimizer Aktion (Python: batch_get_table_optimizer)
- ListTableOptimizerRuns Aktion (Python: list_table_optimizer_runs)
- CreateTableOptimizer Aktion (Python: create_table_optimizer)
- DeleteTableOptimizer Aktion (Python: delete_table_optimizer)
- UpdateTableOptimizer Aktion (Python: update_table_optimizer)
- Crawler- und Classifier-API
  - Classifier-API
    - Datentypen
    - Classifier-Struktur
    - GrokClassifier Struktur
    - XMLClassifier Struktur
    - JsonClassifier Struktur
    - CsvClassifier Struktur
    - CreateGrokClassifierRequest Struktur
    - UpdateGrokClassifierRequest Struktur
    - Anforderungsstruktur erstellen XMLClassifier
    - Struktur der XMLClassifier Anfrage aktualisieren
    - CreateJsonClassifierRequest Struktur
    - UpdateJsonClassifierRequest Struktur
    - CreateCsvClassifierRequest Struktur
    - UpdateCsvClassifierRequest Struktur
    - Operationen
    - CreateClassifier Aktion (Python: create_classifier)
    - DeleteClassifier Aktion (Python: delete_classifier)
    - GetClassifier Aktion (Python: get_classifier)
    - GetClassifiers Aktion (Python: get_classifiers)
    - UpdateClassifier Aktion (Python: update_classifier)
  - Crawler-API
    - Datentypen

- Crawler-Struktur
- Planstruktur
- CrawlerTargets Struktur
- S3Target-Struktur
- DeltaCatalogTarget S3-Struktur
- DeltaDirectTarget S3-Struktur
- JdbcTarget Struktur
- DBTarget Mongo-Struktur
- DBTarget Dynamo-Struktur
- DeltaTarget Struktur
- IcebergTarget Struktur
- HudiTarget Struktur
- CatalogTarget Struktur
- CrawlerMetrics Struktur
- CrawlerHistory Struktur
- CrawlsFilter Struktur
- SchemaChangePolicy Struktur
- LastCrawlInfo Struktur
- RecrawlPolicy Struktur
- LineageConfiguration Struktur
- LakeFormationConfiguration Struktur
- Operationen
- CreateCrawler Aktion (Python: create_crawler)
- DeleteCrawler Aktion (Python: delete_crawler)
- GetCrawler Aktion (Python: get_crawler)
- GetCrawlers Aktion (Python: get_crawlers)
- GetCrawlerMetrics Aktion (Python: get_crawler_metrics)
- UpdateCrawler Aktion (Python: update_crawler)
- StartCrawler Aktion (Python: start_crawler)
- StopCrawler Aktion (Python: stop_crawler)

    - BatchGetCrawlers Aktion (Python: batch_get_crawlers)
    - ListCrawlers Aktion (Python: list_crawlers)
    - ListCrawls Aktion (Python: list_crawls)
- API für Spaltenstatistiken
    - Datentypen
    - ColumnStatisticsTaskRun Struktur
    - ColumnStatisticsTaskSettings Struktur
    - ExecutionAttempt Struktur
    - Operationen
    - StartColumnStatisticsTaskRun Aktion (Python: start_column_statistics_task_run)
    - GetColumnStatisticsTaskRun Aktion (Python: get_column_statistics_task_run)
    - GetColumnStatisticsTaskRuns Aktion (Python: get_column_statistics_task_runs)
    - ListColumnStatisticsTaskRuns Aktion (Python: list_column_statistics_task_runs)
    - StopColumnStatisticsTaskRun Aktion (Python: stop_column_statistics_task_run)
    - CreateColumnStatisticsTaskSettings Aktion (Python: create_column_statistics_task_settings)
    - UpdateColumnStatisticsTaskSettings Aktion (Python: update_column_statistics_task_settings)
    - GetColumnStatisticsTaskSettings Aktion (Python: get_column_statistics_task_settings)
    - DeleteColumnStatisticsTaskSettings Aktion (Python: delete_column_statistics_task_settings)
    - StartColumnStatisticsTaskRunSchedule Aktion (Python: start_column_statistics_task_run_schedule)
    - StopColumnStatisticsTaskRunSchedule Aktion (Python: stop_column_statistics_task_run_schedule)
    - Ausnahmen
    - ColumnStatisticsTaskRunningException Struktur
    - ColumnStatisticsTaskNotRunningException Struktur
    - ColumnStatisticsTaskStoppingException Struktur
    - ColumnStatisticsTaskAutoConcurrencyLimitException Struktur
    - InvalidCatalogSettingException Struktur
- Crawler-Scheduler-API
    - Datentypen

    - Planstruktur
    - Operationen
    - UpdateCrawlerSchedule Aktion (Python: update_crawler_schedule)
    - StartCrawlerSchedule Aktion (Python: start_crawler_schedule)
    - StopCrawlerSchedule Aktion (Python: stop_crawler_schedule)
- Automatisches Generieren einer API für ETL-Skripts
    - Datentypen
    - CodeGenNode Struktur
    - CodeGenNodeArg Struktur
    - CodeGenEdge Struktur
    - Speicherortstruktur
    - CatalogEntry Struktur
    - MappingEntry Struktur
    - Operationen
    - CreateScript Aktion (Python: create_script)
    - GetDataflowGraph Aktion (Python: get_dataflow_graph)
    - GetMapping Aktion (Python: get_mapping)
    - GetPlan Aktion (Python: get_plan)
- Visual Job API
    - Datentypen
    - CodeGenConfigurationNode Struktur
    - JDBCConnectorStruktur der Optionen
    - StreamingDataPreviewOptions Struktur
    - AthenaConnectorSource Struktur
    - JDBCConnectorStruktur der Quelle
    - SparkConnectorSource Struktur
    - CatalogSource Struktur
    - Meine SQLCatalog Quellstruktur
    - SQL CatalogPostgre-Quellstruktur
    - Oracle-Quellstruktur SQLCatalog

- SQLServerCatalogSource Microsoft-Struktur
- CatalogKinesisSource Struktur
- DirectKinesisSource Struktur
- KinesisStreamingSourceOptions Struktur
- CatalogKafkaSource Struktur
- DirectKafkaSource Struktur
- KafkaStreamingSourceOptions Struktur
- RedshiftSource Struktur
- AmazonRedshiftSource Struktur
- AmazonRedshiftNodeData Struktur
- AmazonRedshiftAdvancedOption Struktur
- Optionsstruktur
- CatalogSource S3-Struktur
- SourceAdditionalOptions S3-Struktur
- CsvSource S3-Struktur
- Direkte JDBCSource Struktur
- DirectSourceAdditionalOptions S3-Struktur
- JsonSource S3-Struktur
- ParquetSource S3-Struktur
- DeltaSource S3-Struktur
- CatalogDeltaSource S3-Struktur
- CatalogDeltaSource Struktur
- HudiSource S3-Struktur
- S3-Struktur CatalogHudiSource
- S3-Struktur ExcelSource
- CatalogHudiSource Struktur
- Dynamo-Quellstruktur DBCatalog
- RelationalCatalogSource Struktur
- JDBCConnectorZielstruktur
- SparkConnectorTarget Struktur

- BasicCatalogTarget Struktur
- Meine SQLCatalog Zielstruktur
- SQLCatalogPostgre-Target-Struktur
- Struktur von Oracle SQLCatalog Target
- SQLServerCatalogTarget Microsoft-Struktur
- RedshiftTarget Struktur
- AmazonRedshiftTarget Struktur
- UpsertRedshiftTargetOptions Struktur
- CatalogTarget S3-Struktur
- GlueParquetTarget S3-Struktur
- CatalogSchemaChangePolicy Struktur
- DirectTarget S3-Struktur
- S3-Struktur HudiCatalogTarget
- S3-Struktur HudiDirectTarget
- S3-Struktur DeltaCatalogTarget
- DeltaDirectTarget S3-Struktur
- S3-Struktur HyperDirectTarget
- IcebergDirectTarget S3-Struktur
- DirectSchemaChangePolicy Struktur
- ApplyMapping Struktur
- Mapping-Struktur
- SelectFields Struktur
- DropFields Struktur
- RenameField Struktur
- Spigot-Struktur
- Join-Struktur
- JoinColumn Struktur
- SplitFields Struktur
- SelectFromCollection Struktur
- FillMissingValues Struktur

- Filter-Struktur
- FilterExpression Struktur
- FilterValue Struktur
- CustomCode Struktur
- SparkSQL-Struktur
- SqlAlias Struktur
- DropNullFields Struktur
- NullCheckBoxList Struktur
- NullValueField Struktur
- Datatype-Struktur
- Merge-Struktur
- Union-Struktur
- PIIDetection Struktur
- Aggregierte Struktur
- DropDuplicates Struktur
- GovernedCatalogTarget Struktur
- GovernedCatalogSource Struktur
- AggregateOperation Struktur
- GlueSchema Struktur
- GlueStudioSchemaColumn Struktur
- GlueStudioColumn Struktur
- DynamicTransform Struktur
- TransformConfigParameter Struktur
- EvaluateDataQuality Struktur
- DQResultsPublishingOptions Struktur
- DQStopJobOnFailureOptions Struktur
- EvaluateDataQualityMultiFrame Struktur
- Struktur des Rezepts
- RecipeReference Struktur
- SnowflakeNodeData Struktur

- SnowflakeSource Struktur
- SnowflakeTarget Struktur
- ConnectorDataSource Struktur
- ConnectorDataTarget Struktur
- RecipeStep Struktur
- RecipeAction Struktur
- ConditionExpression Struktur

- Auftrags-API
  - Aufträge
    - Datentypen
    - Auftrags-Struktur
    - ExecutionProperty Struktur
    - NotificationProperty Struktur
    - JobCommand Struktur
    - ConnectionsList Struktur
    - JobUpdate Struktur
    - SourceControlDetails Struktur
    - Operationen
    - CreateJob Aktion (Python: create_job)
    - UpdateJob Aktion (Python: update_job)
    - GetJob Aktion (Python: get_job)
    - GetJobs Aktion (Python: get_jobs)
    - DeleteJob Aktion (Python: delete_job)
    - ListJobs Aktion (Python: list_jobs)
    - BatchGetJobs Aktion (Python: batch_get_jobs)
  - Auftragsausführungen
    - Datentypen
    - JobRun Struktur
    - Vorgängerstruktur
    - JobBookmarkEntry Struktur

    - BatchStopJobRunSuccessfulSubmission Struktur
    - BatchStopJobRunError Struktur
    - NotificationProperty Struktur
    - Operationen
    - StartJobRun Aktion (Python: start_job_run)
    - BatchStopJobRun Aktion (Python: batch_stop_job_run)
    - GetJobRun Aktion (Python: get_job_run)
    - GetJobRuns Aktion (Python: get_job_runs)
    - GetJobBookmark Aktion (Python: get_job_bookmark)
    - GetJobBookmarks Aktion (Python: get_job_bookmarks)
    - ResetJobBookmark Aktion (Python: reset_job_bookmark)
- Auslöser
    - Datentypen
    - Auslöserstruktur
    - TriggerUpdate Struktur
    - Prädikatstruktur
    - Bedingungsstruktur
    - Aktionsstruktur
    - EventBatchingCondition Struktur
    - Operationen
    - CreateTrigger Aktion (Python: create_trigger)
    - StartTrigger Aktion (Python: start_trigger)
    - GetTrigger Aktion (Python: get_trigger)
    - GetTriggers Aktion (Python: get_triggers)
    - UpdateTrigger Aktion (Python: update_trigger)
    - StopTrigger Aktion (Python: stop_trigger)
    - DeleteTrigger Aktion (Python: delete_trigger)
    - ListTriggers Aktion (Python: list_triggers)
    - BatchGetTriggers Aktion (Python: batch_get_triggers)

- Integration APIs in AWS Glue

- Datentypen
- Struktur der Integration
- IntegrationConfig Struktur
- IntegrationPartition Struktur
- IntegrationError Struktur
- IntegrationFilter Struktur
- InboundIntegration Struktur
- SourceProcessingProperties Struktur
- TargetProcessingProperties Struktur
- SourceTableConfig Struktur
- TargetTableConfig Struktur
- Operationen
- CreateIntegration Aktion (Python: create_integration)
- ModifyIntegration Aktion (Python: modify_integration)
- DescribeIntegrations Aktion (Python: describe_integrations)
- DeleteIntegration Aktion (Python: delete_integration)
- DescribeInboundIntegrations Aktion (Python: describe_inbound_integrations)
- CreateIntegrationTableProperties Aktion (Python: create_integration_table_properties)
- UpdateIntegrationTableProperties Aktion (Python: update_integration_table_properties)
- GetIntegrationTableProperties Aktion (Python: get_integration_table_properties)
- DeleteIntegrationTableProperties Aktion (Python: delete_integration_table_properties)
- CreateIntegrationResourceProperty Aktion (Python: create_integration_resource_property)
- UpdateIntegrationResourceProperty Aktion (Python: update_integration_resource_property)
- GetIntegrationResourceProperty Aktion (Python: get_integration_resource_property)
- UntagResource Aktion (Python: untag_resource)
- ListTagsForResource Aktion (Python: list_tags_for_resource)
- Ausnahmen
- ResourceNotFoundException Struktur
- InternalServerException Struktur
- IntegrationAlreadyExistsFault Struktur

- IntegrationConflictOperationFault Struktur
- IntegrationQuotaExceededFault Struktur
- KMSKeyNotAccessibleFault Struktur
- IntegrationNotFoundFault Struktur
- TargetResourceNotFound Struktur
- InvalidIntegrationStateFault Struktur

- Interaktive Sitzungs-API
  - Datentypen
  - Sitzungsstruktur
  - SessionCommand Struktur
  - Statement-Struktur
  - StatementOutput Struktur
  - StatementOutputData Struktur
  - ConnectionsList Struktur
  - Operationen
  - CreateSession Aktion (Python: create_session)
  - StopSession Aktion (Python: stop_session)
  - DeleteSession Aktion (Python: delete_session)
  - GetSession Aktion (Python: get_session)
  - ListSessions Aktion (Python: list_sessions)
  - RunStatement Aktion (Python: run_statement)
  - CancelStatement Aktion (Python: cancel_statement)
  - GetStatement Aktion (Python: get_statement)
  - ListStatements Aktion (Python: list_statements)
- API für Entwicklungsendpunkte
  - Datentypen
  - DevEndpoint Struktur
  - DevEndpointCustomLibraries Struktur
  - Operationen
  - CreateDevEndpoint Aktion (Python: create_dev_endpoint)

- UpdateDevEndpoint Aktion (Python: update_dev_endpoint)
- DeleteDevEndpoint Aktion (Python: delete_dev_endpoint)
- GetDevEndpoint Aktion (Python: get_dev_endpoint)
- GetDevEndpoints Aktion (Python: get_dev_endpoints)
- BatchGetDevEndpoints Aktion (Python: batch_get_dev_endpoints)
- ListDevEndpoints Aktion (Python: list_dev_endpoints)

- Schema Registry
  - Datentypen
  - RegistryId Struktur
  - RegistryListItem Struktur
  - MetadataInfo Struktur
  - OtherMetadataValueListItem Struktur
  - SchemaListItem Struktur
  - SchemaVersionListItem Struktur
  - MetadataKeyValuePair Struktur
  - SchemaVersionErrorItem Struktur
  - ErrorDetails Struktur
  - SchemaVersionNumber Struktur
  - SchemaId Struktur
  - Operationen
  - CreateRegistry Aktion (Python: create_registry)
  - CreateSchema Aktion (Python: create_schema)
  - GetSchema Aktion (Python: get_schema)
  - ListSchemaVersions Aktion (Python: list_schema_versions)
  - GetSchemaVersion Aktion (Python: get_schema_version)
  - GetSchemaVersionsDiff Aktion (Python: get_schema_versions_diff)
  - ListRegistries Aktion (Python: list_registries)
  - ListSchemas Aktion (Python: list_schemas)
  - RegisterSchemaVersion Aktion (Python: register_schema_version)
  - UpdateSchema Aktion (Python: update_schema)

- CheckSchemaVersionValidity Aktion (Python: check_schema_version_validity)
- UpdateRegistry Aktion (Python: update_registry)
- GetSchemaByDefinition Aktion (Python: get_schema_by_definition)
- GetRegistry Aktion (Python: get_registry)
- PutSchemaVersionMetadata Aktion (Python: put_schema_version_metadata)
- QuerySchemaVersionMetadata Aktion (Python: query_schema_version_metadata)
- RemoveSchemaVersionMetadata Aktion (Python: remove_schema_version_metadata)
- DeleteRegistry Aktion (Python: delete_registry)
- DeleteSchema Aktion (Python: delete_schema)
- DeleteSchemaVersions Aktion (Python: delete_schema_versions)

- Workflows
  - Datentypen
  - JobNodeDetails Struktur
  - CrawlerNodeDetails Struktur
  - TriggerNodeDetails Struktur
  - Crawl-Struktur
  - Knotenstruktur
  - Edge-Struktur
  - Workflow-Struktur
  - WorkflowGraph Struktur
  - WorkflowRun Struktur
  - WorkflowRunStatistics Struktur
  - StartingEventBatchCondition Struktur
  - Blueprint-Struktur
  - BlueprintDetails Struktur
  - LastActiveDefinition Struktur
  - BlueprintRun Struktur
  - Operationen
  - CreateWorkflow Aktion (Python: create_workflow)
  - UpdateWorkflow Aktion (Python: update_workflow)

- DeleteWorkflow Aktion (Python: delete_workflow)
- GetWorkflow Aktion (Python: get_workflow)
- ListWorkflows Aktion (Python: list_workflows)
- BatchGetWorkflows Aktion (Python: batch_get_workflows)
- GetWorkflowRun Aktion (Python: get_workflow_run)
- GetWorkflowRuns Aktion (Python: get_workflow_runs)
- GetWorkflowRunProperties Aktion (Python: get_workflow_run_properties)
- PutWorkflowRunProperties Aktion (Python: put_workflow_run_properties)
- CreateBlueprint Aktion (Python: create_blueprint)
- UpdateBlueprint Aktion (Python: update_blueprint)
- DeleteBlueprint Aktion (Python: delete_blueprint)
- ListBlueprints Aktion (Python: list_blueprints)
- BatchGetBlueprints Aktion (Python: batch_get_blueprints)
- StartBlueprintRun Aktion (Python: start_blueprint_run)
- GetBlueprintRun Aktion (Python: get_blueprint_run)
- GetBlueprintRuns Aktion (Python: get_blueprint_runs)
- StartWorkflowRun Aktion (Python: start_workflow_run)
- StopWorkflowRun Aktion (Python: stop_workflow_run)
- ResumeWorkflowRun Aktion (Python: resume_workflow_run)
- Nutzungsprofile
  - Datentypen
  - ProfileConfiguration Struktur
  - ConfigurationObject Struktur
  - UsageProfileDefinition Struktur
  - Operationen
  - CreateUsageProfile Aktion (Python: create_usage_profile)
  - GetUsageProfile Aktion (Python: get_usage_profile)
  - UpdateUsageProfile Aktion (Python: update_usage_profile)
  - DeleteUsageProfile Aktion (Python: delete_usage_profile)
  - ListUsageProfiles Aktion (Python: list_usage_profiles)

- Machine Learning API
  - Datentypen
  - TransformParameters Struktur
  - EvaluationMetrics Struktur
  - MLTransform Struktur
  - FindMatchesParameters Struktur
  - FindMatchesMetrics Struktur
  - ConfusionMatrix Struktur
  - GlueTable Struktur
  - TaskRun Struktur
  - TransformFilterCriteria Struktur
  - TransformSortCriteria Struktur
  - TaskRunFilterCriteria Struktur
  - TaskRunSortCriteria Struktur
  - TaskRunProperties Struktur
  - FindMatchesTaskRunProperties Struktur
  - ImportLabelsTaskRunProperties Struktur
  - ExportLabelsTaskRunProperties Struktur
  - LabelingSetGenerationTaskRunProperties Struktur
  - SchemaColumn Struktur
  - TransformEncryption Struktur
  - MLUserDataEncryption Struktur
  - ColumnImportance Struktur
  - Operationen
  - MLTransform Aktion erstellen (Python: create_ml_transform)
  - MLTransform Aktion aktualisieren (Python: update_ml_transform)
  - MLTransform Aktion löschen (Python: delete_ml_transform)
  - MLTransform Aktion abrufen (Python: get_ml_transform)
  - MLTransforms Aktion abrufen (Python: get_ml_transforms)
  - MLTransforms Aktion auflisten (Python: list_ml_transforms)

- MLEvaluationTaskRun Aktion starten (Python: start_ml_evaluation_task_run)
- MLLabelingSetGenerationTaskRun Aktion starten (Python: start_ml_labeling_set_generation_task_run)
- Aktion „MLTaskAusführen abrufen“ (Python: get_ml_task_run)
- Aktion „Get MLTask Runs“ (Python: get_ml_task_runs)
- Aktion „MLTaskAusführen“ abbrechen (Python: cancel_ml_task_run)
- StartExportLabelsTaskRun Aktion (Python: start_export_labels_task_run)
- StartImportLabelsTaskRun Aktion (Python: start_import_labels_task_run)

- Data-Quality-API
  - Datentypen
  - DataSource Struktur
  - DataQualityRulesetListDetails Struktur
  - DataQualityTargetTable Struktur
  - DataQualityRulesetEvaluationRunDescription Struktur
  - DataQualityRulesetEvaluationRunFilter Struktur
  - DataQualityEvaluationRunAdditionalRunOptions Struktur
  - DataQualityRuleRecommendationRunDescription Struktur
  - DataQualityRuleRecommendationRunFilter Struktur
  - DataQualityResult Struktur
  - DataQualityAnalyzerResult Struktur
  - DataQualityObservation Struktur
  - MetricBasedObservation Struktur
  - DataQualityMetricValues Struktur
  - DataQualityRuleResult Struktur
  - DataQualityResultDescription Struktur
  - DataQualityResultFilterCriteria Struktur
  - DataQualityRulesetFilterCriteria Struktur
  - DataQualityAggregatedMetrics Struktur
  - StatisticAnnotation Struktur
  - TimestampedInclusionAnnotation Struktur

- AnnotationError Struktur
- DatapointInclusionAnnotation Struktur
- StatisticSummaryList Liste
- StatisticSummary Struktur
- RunIdentifier Struktur
- StatisticModelResult Struktur
- Operationen
- StartDataQualityRulesetEvaluationRun Aktion (Python: start_data_quality_ruleset_evaluation_run)
- CancelDataQualityRulesetEvaluationRun Aktion (Python: cancel_data_quality_ruleset_evaluation_run)
- GetDataQualityRulesetEvaluationRun Aktion (Python: get_data_quality_ruleset_evaluation_run)
- ListDataQualityRulesetEvaluationRuns Aktion (Python: list_data_quality_ruleset_evaluation_runs)
- StartDataQualityRuleRecommendationRun Aktion (Python: start_data_quality_rule_recommendation_run)
- CancelDataQualityRuleRecommendationRun Aktion (Python: cancel_data_quality_rule_recommendation_run)
- GetDataQualityRuleRecommendationRun Aktion (Python: get_data_quality_rule_recommendation_run)
- ListDataQualityRuleRecommendationRuns Aktion (Python: list_data_quality_rule_recommendation_runs)
- GetDataQualityResult Aktion (Python: get_data_quality_result)
- BatchGetDataQualityResult Aktion (Python: batch_get_data_quality_result)
- ListDataQualityResults Aktion (Python: list_data_quality_results)
- CreateDataQualityRuleset Aktion (Python: create_data_quality_ruleset)
- DeleteDataQualityRuleset Aktion (Python: delete_data_quality_ruleset)
- GetDataQualityRuleset Aktion (Python: get_data_quality_ruleset)
- ListDataQualityRulesets Aktion (Python: list_data_quality_rulesets)
- UpdateDataQualityRuleset Aktion (Python: update_data_quality_ruleset)
- ListDataQualityStatistics Aktion (Python: list_data_quality_statistics)
- TimestampFilter Struktur

- CreateDataQualityRulesetRequest Struktur
- GetDataQualityRulesetResponse Struktur
- GetDataQualityResultResponse Struktur
- StartDataQualityRuleRecommendationRunRequest Struktur
- GetDataQualityRuleRecommendationRunResponse Struktur
- BatchPutDataQualityStatisticAnnotation Aktion (Python: batch_put_data_quality_statistic_annotation)
- GetDataQualityModel Aktion (Python: get_data_quality_model)
- GetDataQualityModelResult Aktion (Python: get_data_quality_model_result)
- ListDataQualityStatisticAnnotations Aktion (Python: list_data_quality_statistic_annotations)
- PutDataQualityProfileAnnotation Aktion (Python: put_data_quality_profile_annotation)
- API zur Erkennung sensibler Daten
  - Datentypen
  - CustomEntityType Struktur
  - Operationen
  - CreateCustomEntityType Aktion (Python: create_custom_entity_type)
  - DeleteCustomEntityType Aktion (Python: delete_custom_entity_type)
  - GetCustomEntityType Aktion (Python: get_custom_entity_type)
  - BatchGetCustomEntityTypes Aktion (Python: batch_get_custom_entity_types)
  - ListCustomEntityTypes Aktion (Python: list_custom_entity_types)
- Einloggen APIs AWS Glue
  - Datentypen
  - Tag-Struktur
  - Operationen
  - TagResource Aktion (Python: tag_resource)
  - UntagResource Aktion (Python: untag_resource)
  - GetTags Aktion (Python: get_tags)
- Gängige Datentypen
  - Tag-Struktur
  - DecimalNumber Struktur

- ErrorDetail Struktur
- PropertyPredicate Struktur
- ResourceUri Struktur
- ColumnStatistics Struktur
- ColumnStatisticsError Struktur
- ColumnError Struktur
- ColumnStatisticsData Struktur
- BooleanColumnStatisticsData Struktur
- DateColumnStatisticsData Struktur
- DecimalColumnStatisticsData Struktur
- DoubleColumnStatisticsData Struktur
- LongColumnStatisticsData Struktur
- StringColumnStatisticsData Struktur
- BinaryColumnStatisticsData Struktur
- Zeichenfolgemuster

- Ausnahmen
  - AccessDeniedException Struktur
  - AlreadyExistsException Struktur
  - ConcurrentModificationException Struktur
  - ConcurrentRunsExceededException Struktur
  - CrawlerNotRunningException Struktur
  - CrawlerRunningException Struktur
  - CrawlerStoppingException Struktur
  - EntityNotFoundException Struktur
  - FederationSourceException Struktur
  - FederationSourceRetryableException Struktur
  - GlueEncryptionException Struktur
  - IdempotentParameterMismatchException Struktur
  - IllegalWorkflowStateException Struktur
  - InternalServiceException Struktur

- InvalidExecutionEngineException Struktur
- InvalidInputException Struktur
- InvalidStateException Struktur
- InvalidTaskStatusTransitionException Struktur
- JobDefinitionErrorException Struktur
- JobRunInTerminalStateException Struktur
- JobRunInvalidStateTransitionException Struktur
- JobRunNotInTerminalStateException Struktur
- LateRunnerException Struktur
- NoScheduleException Struktur
- OperationTimeoutException Struktur
- ResourceNotReadyException Struktur
- ResourceNumberLimitExceededException Struktur
- SchedulerNotRunningException Struktur
- SchedulerRunningException Struktur
- SchedulerTransitioningException Struktur
- UnrecognizedRunnerException Struktur
- ValidationException Struktur
- VersionMismatchException Struktur

Sicherheit APIs in AWS Glue

Die Sicherheits-API beschreibt die Sicherheitsdatentypen und die API im Zusammenhang mit der Sicherheit in AWS Glue.

Datentypen

- DataCatalogEncryptionSettings Struktur
- EncryptionAtRest Struktur
- ConnectionPasswordEncryption Struktur
- EncryptionConfiguration Struktur
- S3Encryption-Struktur

- CloudWatchEncryption Struktur
- JobBookmarksEncryption Struktur
- SecurityConfiguration Struktur
- GluePolicy Struktur
- DataQualityEncryption Struktur

DataCatalogEncryptionSettings Struktur

Enthält Konfigurationsinformationen zur Aufrechterhaltung der Sicherheit des Data Catalog.

Felder

- `EncryptionAtRest` – Ein EncryptionAtRest-Objekt.

  Gibt die encryption-at-rest Konfiguration für den Datenkatalog an.
- `ConnectionPasswordEncryption` – Ein ConnectionPasswordEncryption-Objekt.

  Wenn der Verbindungspasswortschutz aktiviert ist, verwendet der Data Catalog einen vom Kunden bereitgestellten Schlüssel, um das Passwort als Bestandteil von `CreateConnection` oder `UpdateConnection` zu verschlüsseln, und speichert es in den Verbindungseigenschaften im Feld `ENCRYPTED_PASSWORD`. Sie können die Katalogverschlüsselung oder nur Passwortverschlüsselung aktivieren.

EncryptionAtRest Struktur

Gibt die encryption-at-rest Konfiguration für den Datenkatalog an.

Felder

- `CatalogEncryptionMode` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-KMS-WITH-SERVICE-ROLE="SSEKMSWITHSERVICEROLE"`).

  Der encryption-at-rest Modus für die Verschlüsselung von Datenkatalogdaten.
- `SseAwsKmsKeyId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des AWS KMS Schlüssels, der für die Verschlüsselung im Ruhezustand verwendet werden soll.

- `CatalogEncryptionServiceRole` – UTF-8-Zeichenfolge, die Custom string pattern #51 entspricht.

  Die Rolle, die die Verschlüsselung und Entschlüsselung der Datenkatalogobjekte im Namen des Aufrufers AWS Glue übernimmt.

ConnectionPasswordEncryption Struktur

Die Datenstruktur, die vom Data Catalog verwendet wird, um das Passwort als Bestandteil von `CreateConnection` oder `UpdateConnection` zu verschlüsseln und in den Verbindungseigenschaften im Feld `ENCRYPTED_PASSWORD` zu speichern. Sie können die Katalogverschlüsselung oder nur Passwortverschlüsselung aktivieren.

Wenn eine `CreationConnection` Anfrage eingeht, die ein Passwort enthält, verschlüsselt der Datenkatalog das Passwort zunächst mit Ihrem AWS KMS Schlüssel. Anschließend wird das gesamte Verbindungsobjekt erneut verschlüsselt, wenn die Katalogverschlüsselung ebenfalls aktiviert ist.

Für diese Verschlüsselung müssen Sie AWS KMS Schlüsselberechtigungen festlegen, um den Zugriff auf den Kennwortschlüssel entsprechend Ihren Sicherheitsanforderungen zu aktivieren oder einzuschränken. Möglicherweise möchten Sie einstellen, dass nur Administratoren die Entschlüsselungsberechtigung für den Passwortschlüssel haben.

Felder

- `ReturnConnectionPasswordEncrypted` – Erforderlich: Boolean.

  Wenn das `ReturnConnectionPasswordEncrypted`-Flag auf "true" festgelegt ist, bleiben die Passwörter als Antwort auf `GetConnection` und `GetConnections` verschlüsselt. Diese Verschlüsselung wird unabhängig von der Katalogverschlüsselung wirksam.

- `AwsKmsKeyId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein AWS KMS Schlüssel, der zur Verschlüsselung des Verbindungskennworts verwendet wird.

  Wenn der Verbindungskennwortschutz aktiviert ist, `UpdateConnection` benötigt der `CreateConnection` Anrufer mindestens die `kms:Encrypt` Erlaubnis für den angegebenen AWS KMS Schlüssel, um Passwörter zu verschlüsseln, bevor sie im Datenkatalog gespeichert werden.

Sie können die Verschlüsselungsberechtigung festlegen, um den Zugriff auf den Passwortschlüssel entsprechend Ihren Sicherheitsanforderungen zu aktivieren oder einzuschränken.

EncryptionConfiguration Struktur

Gibt eine Verschlüsselungskonfiguration an.

Felder

- `S3Encryption` – Ein Array mit S3Encryption-Objekten.

  Die Verschlüsselungskonfiguration für Amazon Simple Storage Service (Amazon S3)-Daten.
- `CloudWatchEncryption` – Ein CloudWatchEncryption-Objekt.

  Die Verschlüsselungskonfiguration für Amazon CloudWatch.
- `JobBookmarksEncryption` – Ein JobBookmarksEncryption-Objekt.

  Die Verschlüsselungskonfiguration für Auftragslesezeichen.
- `DataQualityEncryption` – Ein DataQualityEncryption-Objekt.

  Die Verschlüsselungskonfiguration für AWS Glue Data Quality-Ressourcen.

S3Encryption-Struktur

Gibt an, wie Amazon Simple Storage Service (Amazon S3)- Daten verschlüsselt werden sollen.

Felder

- `S3EncryptionMode` – UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-S3="SSES3"`).

  Der Verschlüsselungsmodus, der für Amazon S3-Daten verwendet wird.
- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

  Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels zum Verschlüsseln von Daten.

CloudWatchEncryption Struktur

Gibt an, wie CloudWatch Amazon-Daten verschlüsselt werden sollen.

Felder

- `CloudWatchEncryptionMode` – UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `SSE-KMS="SSEKMS"`).

  Der Verschlüsselungsmodus, der für CloudWatch Daten verwendet werden soll.
- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

  Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels zum Verschlüsseln von Daten.

JobBookmarksEncryption Struktur

Gibt an, wie Auftragslesezeichen-Daten verschlüsselt werden sollen.

Felder

- `JobBookmarksEncryptionMode` – UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `CSE-KMS="CSEKMS"`).

  Der Verschlüsselungsmodus, der für Auftragslesezeichen-Daten verwendet wird.
- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

  Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels zum Verschlüsseln von Daten.

SecurityConfiguration Struktur

Gibt eine Sicherheitskonfiguration an.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen der Sicherheitskonfiguration an.
- `CreatedTimeStamp` – Zeitstempel.

Der Zeitpunkt, zu dem diese Sicherheitskonfiguration erstellt wurde.

- `EncryptionConfiguration` – Ein EncryptionConfiguration-Objekt.

  Die Verschlüsselungskonfiguration, die mit dieser Sicherheitskonfiguration verknüpft ist.

GluePolicy Struktur

Eine Struktur für die Rückgabe einer Ressourcenrichtlinie.

Felder

- `PolicyInJson` – UTF-8-Zeichenfolge, mindestens 2 Bytes lang.

  Enthält das angeforderte Richtliniendokument im JSON-Format.

- `PolicyHash` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Enthält den mit dieser Richtlinie verbundenen Hash-Wert.

- `CreateTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Richtlinie erstellt wurde.

- `UpdateTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Richtlinie zuletzt aktualisiert wurde.

DataQualityEncryption Struktur

Gibt an, wie Data Quality-Ressourcen in Ihrem Konto verschlüsselt werden sollen.

Felder

- `DataQualityEncryptionMode` – UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `SSE-KMS="SSEKMS"`).

  Der Verschlüsselungsmodus, der für die Verschlüsselung von Data Quality-Ressourcen verwendet werden soll. Zu diesen Ressourcen gehören Regelsätze für die Datenqualität, Ergebnisse, Statistiken, Modelle zur Erkennung von Anomalien und Beobachtungen.

Gültige Werte beziehen sich auf `SSEKMS` die Verschlüsselung mit einem vom Kunden verwalteten KMS-Schlüssel oder. `DISABLED`

- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

  Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels zum Verschlüsseln von Daten.

Operationen

- GetDataCatalogEncryptionSettings Aktion (Python: get_data_catalog_encryption_settings)
- PutDataCatalogEncryptionSettings Aktion (Python: put_data_catalog_encryption_settings)
- PutResourcePolicy Aktion (Python: put_resource_policy)
- GetResourcePolicy Aktion (Python: get_resource_policy)
- DeleteResourcePolicy Aktion (Python: delete_resource_policy)
- CreateSecurityConfiguration Aktion (Python: create_security_configuration)
- DeleteSecurityConfiguration Aktion (Python: delete_security_configuration)
- GetSecurityConfiguration Aktion (Python: get_security_configuration)
- GetSecurityConfigurations Aktion (Python: get_security_configurations)
- GetResourcePolicies Aktion (Python: get_resource_policies)

GetDataCatalogEncryptionSettings Aktion (Python: get_data_catalog_encryption_settings)

Ruft die Sicherheitskonfiguration für einen bestimmten Katalog ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, für den die Sicherheitskonfiguration abgerufen werden soll. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS

Antwort

- `DataCatalogEncryptionSettings` – Ein DataCatalogEncryptionSettings-Objekt.

  Die angeforderte Sicherheitskonfiguration.

Fehler

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

PutDataCatalogEncryptionSettings Aktion (Python: put_data_catalog_encryption_settings)

Legt die Sicherheitskonfiguration für einen bestimmten Katalog fest. Nachdem die Konfiguration eingestellt wurde, wird die angegebene Verschlüsselung auf jedes nachfolgende Schreiben des Katalogs angewendet.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, für den die Sicherheitskonfiguration festgelegt werden soll. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DataCatalogEncryptionSettings` – Erforderlich: Ein DataCatalogEncryptionSettings-Objekt.

  Die zu setzende Sicherheitskonfiguration.

Antwort

- Keine Antwortparameter.

Fehler

- `InternalServiceException`

- `InvalidInputException`
- `OperationTimeoutException`

PutResourcePolicy Aktion (Python: put_resource_policy)

Legt die Ressourcenrichtlinie für den Data Catalog für die Zugriffskontrolle fest.

Anforderung

- `PolicyInJson` – Erforderlich: UTF-8-Zeichenfolge, mindestens 2 Bytes lang.

  Enthält das einzurichtende Richtliniendokument im JSON-Format.
- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Nicht verwenden. Nur zur internen Verwendung.
- `PolicyHashCondition` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Hash-Wert, der zurückgegeben wird, wenn die vorherige Richtlinie mit `PutResourcePolicy` festgelegt wurde. Sein Zweck ist es, gleichzeitige Änderungen an einer Richtlinie zu verhindern. Verwenden Sie diesen Parameter nicht, wenn keine vorherige Richtlinie festgelegt wurde.
- `PolicyExistsCondition` – UTF-8-Zeichenfolge (zulässige Werte: `MUST_EXIST` | `NOT_EXIST` | `NONE`).

  Ein Wert von `MUST_EXIST` wird verwendet, um eine Richtlinie zu aktualisieren. Ein Wert von `NOT_EXIST` wird verwendet, um eine neue Richtlinie zu erstellen. Wenn ein Wert von `NONE` oder ein Nullwert verwendet wird, hängt der Aufruf nicht von der Existenz einer Richtlinie ab.
- `EnableHybrid` – UTF-8-Zeichenfolge (zulässige Werte: `TRUE` | `FALSE`).

  Wenn `'TRUE'`, gibt diest an, dass Sie beide Methoden verwenden, um kontenübergreifenden Zugriff auf Data Catalog zu gewähren:
  - Durch direkte Aktualisierung der Ressourcenrichtlinie mit `PutResourePolicy`
  - Durch die Verwendung des Befehls Grant permissions in der AWS Management Console.

  Muss auf `'TRUE'` festgelegt sein, wenn Sie die Managementkonsole bereits verwendet haben, um kontoübergreifenden Zugriff zu gewähren, andernfalls schlägt der Aufruf fehl. Standard ist „FALSE“.

Antwort

- `PolicyHash` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Hash der gerade festgelegten Richtlinie. Dies muss in einem nachfolgenden Aufruf enthalten sein, der diese Richtlinie überschreibt oder aktualisiert.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

GetResourcePolicy Aktion (Python: get_resource_policy)

Ruft eine bestimmte Ressourcenrichtlinie ab.

Anforderung

- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der ARN der AWS Glue Ressource, für die die Ressourcenrichtlinie abgerufen werden soll. Erfolgt keine Angabe, wird die Ressourcenrichtlinie für Data Catalog zurückgegeben. Verwenden von `GetResourcePolicies`, um alle vorhandenen Ressourcenrichtlinien anzuzeigen. Weitere Informationen finden Sie unter AWS Glue Ressource angeben ARNs.

Antwort

- `PolicyInJson` – UTF-8-Zeichenfolge, mindestens 2 Bytes lang.

  Enthält das angeforderte Richtliniendokument im JSON-Format.
- `PolicyHash` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Enthält den mit dieser Richtlinie verbundenen Hash-Wert.

- `CreateTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Richtlinie erstellt wurde.

- `UpdateTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Richtlinie zuletzt aktualisiert wurde.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

DeleteResourcePolicy Aktion (Python: delete_resource_policy)

Löscht eine angegebene Richtlinie.

Anforderung

- `PolicyHashCondition` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Hash-Wert wird zurückgegeben, wenn diese Richtlinie festgelegt wurde.

- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der ARN der AWS Glue Ressource für die zu löschende Ressourcenrichtlinie.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

CreateSecurityConfiguration Aktion (Python: create_security_configuration)

Erstellt eine neue Sicherheitskonfiguration. Eine Sicherheitskonfiguration ist eine Reihe von Sicherheitseigenschaften, die von AWS Glue verwendet werden können. Sie können eine Sicherheitskonfiguration verwenden, um Daten im Ruhezustand zu verschlüsseln. Informationen zur Verwendung von Sicherheitskonfigurationen finden Sie unter Verschlüsselung von Daten AWS Glue, die von Crawlern, Jobs und Entwicklungsendpunkten geschrieben wurden.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name für die neue Sicherheitskonfiguration.
- `EncryptionConfiguration` – Erforderlich: Ein EncryptionConfiguration-Objekt.

  Die Verschlüsselungskonfiguration für die neue Sicherheitskonfiguration.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name, der der neuen Sicherheitskonfiguration zugewiesen wurde.
- `CreatedTimestamp` – Zeitstempel.

  Der Zeitpunkt, zu dem die neue Sicherheitskonfiguration erstellt wurde.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteSecurityConfiguration Aktion (Python: delete_security_configuration)

Löscht eine angegebene Sicherheitskonfiguration.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu löschenden Sicherheitskonfiguration.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfiguration Aktion (Python: get_security_configuration)

Ruft eine angegebene Sicherheitskonfiguration ab.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der abzurufenden Sicherheitskonfiguration.

Antwort

- `SecurityConfiguration` – Ein SecurityConfiguration-Objekt.

  Die angeforderte Sicherheitskonfiguration.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfigurations Aktion (Python: get_security_configurations)

Ruft eine Liste aller Sicherheitskonfigurationen ab.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `SecurityConfigurations` – Ein Array mit SecurityConfiguration-Objekten.

  Eine Liste der Sicherheitskonfigurationen.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn mehr Sicherheitskonfigurationen zurückgegeben werden müssen.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetResourcePolicies Aktion (Python: get_resource_policies)

Ruft die Ressourcenrichtlinien ab, die für einzelne Ressourcen bei kontoübergreifenden Berechtigungsgewährungen festgelegt wurden. AWS Resource Access Manager Ruft auch die Ressourcenrichtlinie für den Data Catalog ab.

Wenn Sie die Metadatenverschlüsselung in den Datenkatalogeinstellungen aktiviert haben und keine Berechtigung für den AWS KMS Schlüssel haben, kann der Vorgang die Datenkatalog-Ressourcenrichtlinie nicht zurückgeben.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.

Antwort

- `GetResourcePoliciesResponseList` – Ein Array mit GluePolicy-Objekten.

  Eine Liste der einzelnen Ressourcenrichtlinien und der Ressourcenrichtlinie auf Kontoebene.
- `NextToken` – UTF-8-Zeichenfolge.

Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Ressourcenrichtlinie nicht enthält.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

API für Katalogobjekte

Die API für Katalogobjekte beschreibt die Datentypen und die API im Zusammenhang mit der Arbeit mit Katalogen in AWS Glue.

Themen

- Kataloge-API
- Datenbank-API
- Tabellen-API
- Partitions-API
- Verbindungs-API
- Benutzerdefinierte Funktion (API)
- Einen Athena Katalog importieren nach AWS Glue

Kataloge-API

Die Catalogs API beschreibt das APIs Erstellen, Löschen, Suchen, Aktualisieren und Auflisten von Katalogen.

Datentypen

- Struktur des Katalogs
- CatalogInput Struktur

- TargetRedshiftCatalog Struktur
- CatalogProperties Struktur
- CatalogPropertiesOutput Struktur
- DataLakeAccessProperties Struktur
- DataLakeAccessPropertiesOutput Struktur
- FederatedCatalog Struktur

Struktur des Katalogs

Das Katalogobjekt stellt eine logische Gruppierung von Datenbanken im AWS Glue Datenkatalog oder in einer Verbundquelle dar. Sie können jetzt einen Redshift-Verbundkatalog oder einen Katalog mit Ressourcenlinks zu Redshift-Datenbanken in einem anderen Konto oder einer anderen Region erstellen.

Felder

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs. Um Zugriff auf den Standardkatalog zu gewähren, sollte dieses Feld nicht angegeben werden.
- `Name`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 64 Byte lang, entspricht. Custom string pattern #25

  Der Name des Katalogs. Kann nicht mit der Konto-ID identisch sein.
- `ResourceArn` – UTF-8-Zeichenfolge.

  Der Amazon-Ressourcenname (ARN), der der Katalogressource zugewiesen wurde.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibungszeichenfolge, nicht mehr als 2048 Byte lang, entspricht dem mehrzeiligen Zeichenkettenmuster der URI-Adresse. Eine Beschreibung des Katalogs.
- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

Ein Map-Array von Schlüssel-Wert-Paaren, die Parameter und Eigenschaften des Katalogs definieren.

- `CreateTime` – Zeitstempel.

  Der Zeitpunkt, zu dem der Katalog erstellt wurde.

- `UpdateTime` – Zeitstempel.

  Der Zeitpunkt, zu dem der Katalog zuletzt aktualisiert wurde.

- `TargetRedshiftCatalog` – Ein TargetRedshiftCatalog-Objekt.

  Ein `TargetRedshiftCatalog` Objekt, das einen Zielkatalog für die Verknüpfung von Datenbankressourcen beschreibt.

- `FederatedCatalog` – Ein FederatedCatalog-Objekt.

  Ein `FederatedCatalog` Objekt, das auf eine Entität außerhalb des AWS Glue Datenkatalogs verweist.

- `CatalogProperties` – Ein CatalogPropertiesOutput-Objekt.

  Ein `CatalogProperties` Objekt, das Eigenschaften für den Zugriff auf Data Lake und andere benutzerdefinierte Eigenschaften angibt.

- `CreateTableDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

  Ein Array von `PrincipalPermissions`-Objekten. Erstellt eine Reihe von Standardberechtigungen für die Tabelle (n) für Prinzipale. Wird verwendet von AWS Lake Formation. Wird im normalen AWS Glue Betriebsablauf nicht verwendet.

- `CreateDatabaseDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

  Ein Array von `PrincipalPermissions`-Objekten. Erstellt eine Reihe von Standardberechtigungen für die Datenbank (en) für Prinzipale. Wird verwendet von AWS Lake Formation. Wird im normalen AWS Glue Betriebsablauf nicht verwendet.

- `AllowFullTableExternalDataAccess` – UTF-8-Zeichenfolge (zulässige Werte: `True` | `False`).

  Ermöglicht Engines von Drittanbietern den Zugriff auf Daten an Amazon S3 Standorten, die bei Lake Formation registriert sind.

CatalogInput Struktur

Eine Struktur, die Katalogeigenschaften beschreibt.

Felder

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibungszeichenfolge, nicht mehr als 2048 Byte lang, entspricht dem mehrzeiligen Zeichenkettenmuster der URI-Adresse. Eine Beschreibung des Katalogs.

- `FederatedCatalog` – Ein FederatedCatalog-Objekt.

  Ein `FederatedCatalog`-Objekt. Eine `FederatedCatalog` Struktur, die auf eine Entität außerhalb des AWS Glue Datenkatalogs verweist, z. B. eine Redshift-Datenbank.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Ein Map-Array von Schlüssel-Wert-Paaren, die die Parameter und Eigenschaften des Katalogs definieren.

- `TargetRedshiftCatalog` – Ein TargetRedshiftCatalog-Objekt.

  Ein `TargetRedshiftCatalog` Objekt, das einen Zielkatalog für die Ressourcenverknüpfung beschreibt.

- `CatalogProperties` – Ein CatalogProperties-Objekt.

  Ein `CatalogProperties` Objekt, das Data Lake-Zugriffseigenschaften und andere benutzerdefinierte Eigenschaften angibt.

- `CreateTableDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

  Ein Array von `PrincipalPermissions`-Objekten. Erstellt eine Reihe von Standardberechtigungen für die Tabelle (n) für Prinzipale. Wird verwendet von AWS Lake Formation. Sollte normalerweise explizit als leere Liste festgelegt werden.

- `CreateDatabaseDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

Ein Array von `PrincipalPermissions`-Objekten. Erstellt eine Reihe von Standardberechtigungen für die Datenbank (en) für Prinzipale. Wird verwendet von AWS Lake Formation. Sollte normalerweise explizit als leere Liste festgelegt werden.

- `AllowFullTableExternalDataAccess` – UTF-8-Zeichenfolge (zulässige Werte: `True` | `False`).

  Ermöglicht Engines von Drittanbietern den Zugriff auf Daten an Amazon S3 Standorten, die bei Lake Formation registriert sind.

TargetRedshiftCatalog Struktur

Eine Struktur, die einen Zielkatalog für die Verknüpfung von Ressourcen beschreibt.

Felder

- `CatalogArn` – Erforderlich: UTF-8-Zeichenfolge.

  Der Amazon-Ressourcenname (ARN) der Katalogressource.

CatalogProperties Struktur

Eine Struktur, die Eigenschaften für den Zugriff auf Data Lake und andere benutzerdefinierte Eigenschaften festlegt.

Felder

- `DataLakeAccessProperties` – Ein DataLakeAccessProperties-Objekt.

  Ein `DataLakeAccessProperties` Objekt, das Eigenschaften zur Konfiguration des Data Lake-Zugriffs für Ihre Katalogressource im AWS Glue Datenkatalog angibt.
- `CustomProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Zusätzliche Schlüsselwerteigenschaften für den Katalog, z. B. Optimierungen der Spaltenstatistiken.

CatalogPropertiesOutput Struktur

Eigenschaftsattribute, die Konfigurationseigenschaften für die Katalogressource enthalten.

Felder

- `DataLakeAccessProperties` – Ein DataLakeAccessPropertiesOutput-Objekt.

  Ein `DataLakeAccessProperties` Objekt mit Eingabeeigenschaften zur Konfiguration des Data Lake-Zugriffs für Ihre Katalogressource im AWS Glue Datenkatalog.
- `CustomProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Zusätzliche Schlüsselwerteigenschaften für den Katalog, z. B. Optimierungen der Spaltenstatistiken.

DataLakeAccessProperties Struktur

Geben Sie Eigenschaften ein, um den Data Lake-Zugriff für Ihre Katalogressource im AWS Glue Datenkatalog zu konfigurieren.

Felder

- `DataLakeAccess` – Boolesch.

  Aktiviert oder deaktiviert den Data-Lake-Zugriff für Apache Spark-Anwendungen, die über eine Nicht-Redshift-Engine wie Amazon Athena, Amazon EMR oder ETL auf Amazon Redshift-Datenbanken im Data Catalog zugreifen. AWS Glue
- `DataTransferRole` – UTF-8-Zeichenfolge, die Custom string pattern #51 entspricht.

  Eine Rolle, die AWS Glue für die Übertragung von Daten aus dem Staging-Bucket während into/out einer Abfrage übernommen wird.
- `KmsKey` – UTF-8-Zeichenfolge.

  Ein Verschlüsselungsschlüssel, der für den Staging-Bucket verwendet wird, der zusammen mit dem Katalog erstellt wird.

- `CatalogType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt einen Verbundkatalogtyp für die systemeigene Katalogressource an. Der derzeit unterstützte Typ ist `aws:redshift`.

DataLakeAccessPropertiesOutput Struktur

Die Ausgabeeigenschaften der Data Lake-Zugriffskonfiguration für Ihre Katalogressource im AWS Glue Datenkatalog.

Felder

- `DataLakeAccess` – Boolesch.

  Aktiviert oder deaktiviert den Data Lake-Zugriff für Apache Spark-Anwendungen, die auf Amazon Redshift Redshift-Datenbanken im Datenkatalog zugreifen.
- `DataTransferRole` – UTF-8-Zeichenfolge, die Custom string pattern #51 entspricht.

  Eine Rolle, die AWS Glue für die Übertragung into/out von Daten aus dem Staging-Bucket während einer Abfrage übernommen wird.
- `KmsKey` – UTF-8-Zeichenfolge.

  Ein Verschlüsselungsschlüssel, der für den Staging-Bucket verwendet wird, der zusammen mit dem Katalog erstellt wird.
- `ManagedWorkgroupName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der verwaltete Redshift Serverless Computename, der für Ihre Katalogressource erstellt wurde.
- `ManagedWorkgroupStatus` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der verwaltete Redshift Serverless Compute-Status.
- `RedshiftDatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Standardname der Redshift-Datenbankressource in der verwalteten Recheneinheit.

- `StatusMessage` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Nachricht, die detailliertere Informationen zum Status der verwalteten Arbeitsgruppe enthält.
- `CatalogType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt einen Verbundkatalogtyp für die systemeigene Katalogressource an. Der derzeit unterstützte Typ ist `aws:redshift`.

FederatedCatalog Struktur

Ein Katalog, der auf eine Entität außerhalb des AWS Glue Datenkatalogs verweist.

Felder

- `Identifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für den Verbundkatalog.
- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung zu einer externen Datenquelle, z. B. einem RedShift-Verbundkatalog.
- `ConnectionType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Verbindungstyp, der für den Zugriff auf den Verbundkatalog verwendet wird, wobei das Protokoll oder die Methode für die Verbindung mit der externen Datenquelle angegeben wird.

Operationen

- CreateCatalog Aktion (Python: create_catalog)
- UpdateCatalog Aktion (Python: update_catalog)
- DeleteCatalog Aktion (Python: delete_catalog)
- GetCatalog Aktion (Python: get_catalog)
- GetCatalogs Aktion (Python: get_catalogs)

CreateCatalog Aktion (Python: create_catalog)

Erstellt einen neuen Katalog im AWS Glue Datenkatalog.

Anforderung

- `Name`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 64 Byte lang, entspricht. Custom string pattern #25

  Der Name des zu erstellenden Katalogs.
- `CatalogInput` – Erforderlich: Ein CatalogInput-Objekt.

  Ein `CatalogInput` Objekt, das die Metadaten für den Katalog definiert.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paaren. Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist. Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist. Die Tags, die Sie dem Katalog zuweisen.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

- `ConcurrentModificationException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederatedResourceAlreadyExistsException`
- `FederationSourceException`

UpdateCatalog Aktion (Python: update_catalog)

Aktualisiert die Eigenschaften eines vorhandenen Katalogs im AWS Glue Datenkatalog.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs.
- `CatalogInput` – Erforderlich: Ein CatalogInput-Objekt.

  Ein `CatalogInput` Objekt, das die neuen Eigenschaften eines vorhandenen Katalogs angibt.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `AccessDeniedException`
- `FederationSourceException`

DeleteCatalog Aktion (Python: delete_catalog)

Entfernt den angegebenen Katalog aus dem AWS Glue Datenkatalog.

Nach Abschluss dieses Vorgangs haben Sie keinen Zugriff mehr auf die Datenbanken, Tabellen (und alle Tabellenversionen und Partitionen, die möglicherweise zu den Tabellen gehören) und die benutzerdefinierten Funktionen im gelöschten Katalog. AWS Glue löscht diese „verwaisten“ Ressourcen nach eigenem Ermessen asynchron und rechtzeitig.

Um sicherzustellen, dass alle zugehörigen Ressourcen sofort gelöscht werden, bevor der `DeleteCatalog` Vorgang aufgerufen wird, verwenden Sie `DeleteTableVersion` `DeletePartition` (oder `BatchDeleteTableVersionBatchDeletePartition`), `DeleteTable` (oder`BatchDeleteTable`), `DeleteUserDefinedFunction` und `DeleteDatabase` löschen Sie alle Ressourcen, die zum Katalog gehören.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `AccessDeniedException`
- `FederationSourceException`

GetCatalog Aktion (Python: get_catalog)

Der Name des abzurufenden Katalogs. Das sollte alles in Kleinbuchstaben geschrieben sein.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des übergeordneten Katalogs, in dem sich der Katalog befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Kontonummer verwendet.

Antwort

- `Catalog` – Ein Katalog-Objekt.

  Ein `Catalog`-Objekt. Die Definition des angegebenen Katalogs im AWS Glue Datenkatalog.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetCatalogs Aktion (Python: get_catalogs)

Ruft alle Kataloge ab, die in einem Katalog im Datenkatalog definiert sind. AWS Glue Für einen Redshift-Verbundkatalog gibt dieser Vorgang die Liste der Kataloge zurück, die Redshift-Datenbanken im Redshift-Namespace-Katalog zugeordnet sind.

Anforderung

- `ParentCatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des übergeordneten Katalogs, in dem sich der Katalog befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Kontonummer verwendet.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl von Katalogen, die in einer Antwort zurückgegeben werden können.

- `Recursive` – Boolesch.

  Ob alle Kataloge in der Kataloghierarchie aufgelistet werden sollen, beginnend mit. `ParentCatalogId` Die Standardeinstellung ist. `false` Wenn`true`, werden alle Katalogobjekte in der `ParentCatalogID` Hierarchie in der Antwort aufgezählt.

- `IncludeRoot` – Boolesch.

  Ob der Standardkatalog für das Konto und die Region in der Antwort aufgeführt werden soll. Standardeinstellung: `false`. Wenn `true` und`ParentCatalogId = NULL | AWS Account ID`, werden alle Kataloge und der Standardkatalog in der Antwort aufgeführt.

  Wenn der nicht gleich Null `ParentCatalogId` ist und dieses Attribut als `false` oder übergeben wird`true`, wird ein `InvalidInputException` ausgelöst.

Antwort

- `CatalogList` – Erforderlich: Ein Array mit Katalog-Objekten.

  Ein Array von `Catalog`-Objekten. Eine Liste von `Catalog` Objekten aus dem angegebenen übergeordneten Katalog.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

Datenbank-API

Die Datenbank-API beschreibt Datenbankdatentypen und umfasst die API zum Erstellen, Löschen, Auffinden, Aktualisieren und Auflisten von Datenbanken.

Datentypen

- Database-Struktur
- DatabaseInput Struktur
- PrincipalPermissions Struktur
- DataLakePrincipal Struktur
- DatabaseIdentifier Struktur
- FederatedDatabase Struktur

Database-Struktur

Das `Database`-Objekt stellt eine logische Gruppierung von Tabellen dar, die sich in einem Hive-Metastore oder einem RDBMS befinden können.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Name der Datenbank. Für Hive-Kompatibilität wird dieser beim Speichern wie Kleinbuchstaben behandelt.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Datenbank.

- `LocationUri` – Uniform Resource Identifier (uri), nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum URI address multi-line string pattern.

  Die Speicheradresse der Datenbank (z. B. HDFS-Pfad).

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren Parameter und Eigenschaften der Datenbank.

- `CreateTime` – Zeitstempel.

  Der Zeitpunkt, zu dem die Metadatenbank im Katalog angelegt wurde.

- `CreateTableDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

  Erstellt einen Satz von Standardberechtigungen für die Tabelle für Prinzipale. Benutzt von AWS Lake Formation. Wird im normalen AWS Glue Betrieb nicht verwendet.

- `TargetDatabase` – Ein DatabaseIdentifier-Objekt.

  Eine `DatabaseIdentifier`-Struktur, die eine Zieldatenbank für die Verknüpfung von Ressourcen beschreibt.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist.

- `FederatedDatabase` – Ein FederatedDatabase-Objekt.

  Eine `FederatedDatabase`-Struktur, die auf eine Entität außerhalb von AWS Glue Data Catalog verweist.

DatabaseInput Struktur

Die Struktur, die zum Anlegen oder Aktualisieren einer Datenbank verwendet wird.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name der Datenbank. Für Hive-Kompatibilität wird dieser beim Speichern wie Kleinbuchstaben behandelt.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Datenbank.

- `LocationUri` – Uniform Resource Identifier (uri), nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum URI address multi-line string pattern.

  Die Speicheradresse der Datenbank (z. B. HDFS-Pfad).

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren Parameter und Eigenschaften der Datenbank.

  Diese Schlüssel-Wert-Paare definieren Parameter und Eigenschaften der Datenbank.

- `CreateTableDefaultPermissions` – Ein Array mit PrincipalPermissions-Objekten.

  Erstellt einen Satz von Standardberechtigungen für die Tabelle für Prinzipale. Benutzt von AWS Lake Formation. Wird im normalen AWS Glue Betrieb nicht verwendet.

- `TargetDatabase` – Ein DatabaseIdentifier-Objekt.

  Eine `DatabaseIdentifier`-Struktur, die eine Zieldatenbank für die Verknüpfung von Ressourcen beschreibt.

- `FederatedDatabase` – Ein FederatedDatabase-Objekt.

Eine `FederatedDatabase`-Struktur, die auf eine Entität außerhalb von AWS Glue Data Catalog verweist.

PrincipalPermissions Struktur

Berechtigungen, die einem Prinzipal erteilt wurden.

Felder

- `Principal` – Ein DataLakePrincipal-Objekt.

  Der Prinzipal, dem Berechtigungen erteilt werden.
- `Permissions` – Ein UTF-8-Zeichenfolgen-Array.

  Die Berechtigungen, die dem Prinzipal gewährt werden.

DataLakePrincipal Struktur

Der AWS Lake Formation Schulleiter.

Felder

- `DataLakePrincipalIdentifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang.

  Eine Kennung für den AWS Lake Formation Principal.

DatabaseIdentifier Struktur

Eine Struktur, die eine Zieldatenbank für die Verknüpfung von Ressourcen beschreibt.

Felder

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist.
- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Katalogdatenbank.

- `Region` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Region der Zieldatenbank.

FederatedDatabase Struktur

Eine Datenbank, die auf eine Entität außerhalb der AWS Glue Data Catalog verweist.

Felder

- `Identifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Verbunddatenbank.

- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung zum externen Metaspeicher.

- `ConnectionType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Verbindungstyp, der für den Zugriff auf die Verbunddatenbank verwendet wird, z. B. JDBC, ODBC oder andere unterstützte Verbindungsprotokolle.

Operationen

- CreateDatabase Aktion (Python: create_database)
- UpdateDatabase Aktion (Python: update_database)
- DeleteDatabase Aktion (Python: delete_database)
- GetDatabase Aktion (Python: get_database)
- GetDatabases Aktion (Python: get_databases)

CreateDatabase Aktion (Python: create_database)

Erstellt eine neue Datenbank in einem Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank erstellt wird. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseInput` – Erforderlich: Ein DatabaseInput-Objekt.

  Die Metadaten für die Datenbank.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags, die Sie der Datenbank zuweisen.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `FederatedResourceAlreadyExistsException`

- `FederationSourceException`
- `FederationSourceRetryableException`

UpdateDatabase Aktion (Python: update_database)

Aktualisiert eine bestehende Datenbankdefinition in einem Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Metadaten-Datenbank gespeichert ist. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, die im Katalog aktualisiert werden soll. Für Hive-Kompatibilität wird er in Kleinbuchstaben gespeichert.
- `DatabaseInput` – Erforderlich: Ein DatabaseInput-Objekt.

  Ein `DatabaseInput`-Objekt, das die neue Definition der Metadaten-Datenbank im Katalog angibt.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `FederationSourceException`

- `FederationSourceRetryableException`
- `AlreadyExistsException`

DeleteDatabase Aktion (Python: delete_database)

Entfernt eine angegebene Datenbank aus einem Data Catalog.

> **Note**
>
> Nach Abschluss dieses Vorgangs haben Sie keinen Zugriff mehr auf die Tabellen (und alle Tabellenversionen und Partitionen, die möglicherweise zu den Tabellen gehören) und die benutzerdefinierten Funktionen in der gelöschten Datenbank. AWS Glue löscht diese „verwaisten“ Ressourcen nach Ermessen des Dienstes asynchron und rechtzeitig.
> Um die sofortige Löschung aller damit verbundenen Ressourcen zu gewährleisten, bevor Sie `DeleteDatabase` aufrufen, verwenden Sie `DeleteTableVersion` oder `BatchDeleteTableVersion`, `DeletePartition` oder `BatchDeletePartition`, `DeleteUserDefinedFunction` und `DeleteTable` oder `BatchDeleteTable`, um alle Ressourcen zu löschen, die zur Datenbank gehören.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu löschenden Datenbank. Für Hive-Kompatibilität muss dieser vollständig aus Kleinbuchstaben bestehen.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetDatabase Aktion (Python: get_database)

Ruft die Definition einer angegebenen Datenbank ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, die abgerufen werden soll. Für Hive-Kompatibilität sollte dieser vollständig aus Kleinbuchstaben bestehen.

Antwort

- `Database` – Ein Datenbank-Objekt.

  Die Definition der angegebenen Datenbank im Data Catalog.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetDatabases Aktion (Python: get_databases)

Ruft alle Datenbanken ab, die in einem bestimmten Data Catalog definiert sind.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, aus dem abgerufen werden soll `Databases`. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Die maximale Anzahl der Datenbanken, die in einer Antwort zurückgegeben werden können.
- `ResourceShareType` – UTF-8-Zeichenfolge (zulässige Werte: `FOREIGN` | `ALL` | `FEDERATED`).

  Hier können Sie angeben, dass Sie sich die für Ihr Konto freigegebenen Datenbanken auflisten lassen möchten. Die zulässigen Werte sind `FEDERATED`, `FOREIGN` oder `ALL`.
  - Wenn auf `FEDERATED` festgelegt, werden die Verbunddatenbanken (die auf eine externe Entität verweisen) aufgelistet, die für Ihr Konto freigegeben sind.
  - Bei Auswahl von `FOREIGN` werden die Datenbanken aufgelistet, die für Ihr Konto freigegeben wurden.
  - Bei Auswahl von `ALL` werden die Datenbanken aufgelistet, die für Ihr Konto freigegeben sind, sowie die Datenbanken in Ihrem lokalen Konto.
- `AttributesToGet` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die vom `GetDatabases` Aufruf zurückgegebenen Datenbankfelder an. Dieser Parameter akzeptiert keine leere Liste. Die Anfrage muss die enthaltenNAME.

Antwort

- `DatabaseList` – Erforderlich: Ein Array mit Datenbank-Objekten.

  Eine Liste von `Database`-Objekten aus dem angegebenen Katalog.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

Tabellen-API

Die Tabellen-API beschreibt Datentypen und Operationen im Zusammenhang mit Tabellen.

Datentypen

- Tabellenstruktur
- TableInput Struktur
- FederatedTable Struktur
- Spaltenstruktur
- StorageDescriptor Struktur
- SchemaReference Struktur
- SerDeInfo Struktur
- Order-Struktur
- SkewedInfo Struktur

- TableVersion Struktur
- TableError Struktur
- TableVersionError Struktur
- SortCriterion Struktur
- TableIdentifier Struktur
- KeySchemaElement Struktur
- PartitionIndex Struktur
- PartitionIndexDescriptor Struktur
- BackfillError Struktur
- IcebergInput Struktur
- OpenTableFormatInput Struktur
- ViewDefinition Struktur
- ViewDefinitionInput Struktur
- ViewRepresentation Struktur
- ViewRepresentationInput Struktur
- UpdateOpenTableFormatInput Struktur
- UpdateIcebergInput Struktur
- CreateIcebergTableInput Struktur
- UpdateIcebergTableInput Struktur
- IcebergSortOrder Struktur
- IcebergSortField Struktur
- IcebergPartitionSpec Struktur
- IcebergPartitionField Struktur
- IcebergSchema Struktur
- IcebergStructField Struktur
- IcebergTableUpdate Struktur

Tabellenstruktur

Stellt eine Sammlung zusammengehöriger Daten organisiert in Spalten und Zeilen dar.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität muss dieser vollständig aus Kleinbuchstaben bestehen.

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabellenmetadaten befinden. Für Hive-Kompatibilität muss dieser vollständig aus Kleinbuchstaben bestehen.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Beschreibung der Tabelle.

- `Owner` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Eigentümer der Tabelle.

- `CreateTime` – Zeitstempel.

  Die Uhrzeit, wann die Tabellendefinition im Data Catalog erstellt wurde.

- `UpdateTime` – Zeitstempel.

  Der letzte Zeitpunkt, an dem die Tabelle aktualisiert wurde.

- `LastAccessTime` – Zeitstempel.

  Der letzte Zeitpunkt, an dem auf die Tabelle zugegriffen wurde. Dieser stammt in der Regel aus HDFS und ist möglicherweise nicht zuverlässig.

- `LastAnalyzedTime` – Zeitstempel.

  Letzter Zeitpunkt der Berechnung der Spaltenstatistiken für diese Tabelle.

- `Retention` – Zahl (Ganzzahl), nicht mehr als Keine.

  Aufbewahrungsdauer für diese Tabelle.

- `StorageDescriptor` – Ein StorageDescriptor-Objekt.

Eine Speicherbeschreibung, die Informationen zur physischen Speicherung dieser Tabelle enthält.

- `PartitionKeys` – Ein Array mit Spalte-Objekten.

  Eine Liste der Spalten, nach denen die Tabelle partitioniert ist. Es werden nur primitive Typen als Partitionsschlüssel unterstützt.

  Wenn Sie eine Tabelle erstellen, die von Amazon Athena verwendet wird, und keine `partitionKeys` angeben, müssen Sie mindestens den Wert der `partitionKeys` auf eine leere Liste festlegen. Zum Beispiel:

  ```
  "PartitionKeys": []
  ```

- `ViewOriginalText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Für Apache-Hive-Kompatibilität enthalten. Wird im normalen AWS Glue Betrieb nicht verwendet. Wenn es sich bei der Tabelle um eine bestimmte Athena Konfiguration handelt`VIRTUAL_VIEW`, die in Base64 codiert ist.

- `ViewExpandedText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Für Apache-Hive-Kompatibilität enthalten. Wird im normalen Betriebsablauf nicht verwendet. AWS Glue

- `TableType` – UTF-8-Zeichenfolge, nicht mehr als 255 Bytes lang.

  Der Typ dieser Tabelle. AWS Glue erstellt Tabellen mit dem `EXTERNAL_TABLE` Typ. Andere Dienste, wie z. B. Athena, können Tabellen mit zusätzlichen Tabellentypen erstellen.

  AWS Glue verwandte Tabellentypen:

  EXTERNAL_TABLE

  Hive-kompatibles Attribut – zeigt eine nicht von Hive verwaltete Tabelle an.

  GOVERNED

  Wird verwendet von AWS Lake Formation. Der AWS Glue Datenkatalog versteht es`GOVERNED`.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

Diese Schlüssel-Wert-Paare definieren die der Tabelle zugeordneten Eigenschaften.

- `CreatedBy` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Person oder Entität, die die Tabelle erstellt hat.

- `IsRegisteredWithLakeFormation` – Boolesch.

  Gibt an, ob die Tabelle bei registriert wurde AWS Lake Formation.

- `TargetTable` – Ein TableIdentifier-Objekt.

  Eine `TableIdentifier`-Struktur, die eine Zieltabelle für die Verknüpfung von Ressourcen beschreibt.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Tabelle gespeichert ist.

- `VersionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Tabellenversion.

- `FederatedTable` – Ein FederatedTable-Objekt.

  Eine `FederatedTable`-Struktur, die auf eine Entität außerhalb von AWS Glue Data Catalog verweist.

- `ViewDefinition` – Ein ViewDefinition-Objekt.

  Eine Struktur, die alle Informationen enthält, die die Ansicht definieren, einschließlich des Dialekts oder der Dialekte für die Ansicht und der Abfrage.

- `IsMultiDialectView` – Boolesch.

  Gibt an, ob die Ansicht die SQL-Dialekte einer oder mehrerer verschiedener Abfrage-Engines unterstützt und daher von diesen Engines gelesen werden kann.

TableInput Struktur

Eine Struktur, die Sie nutzen können, um eine Tabelle zu definieren.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität wird dieser beim Speichern wie Kleinbuchstaben behandelt.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Beschreibung der Tabelle.

- `Owner` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Eigentümer der Tabelle. Für Apache-Hive-Kompatibilität enthalten. Wird im normalen AWS Glue Betriebsablauf nicht verwendet.

- `LastAccessTime` – Zeitstempel.

  Der letzte Zeitpunkt, an dem auf die Tabelle zugegriffen wurde.

- `LastAnalyzedTime` – Zeitstempel.

  Letzter Zeitpunkt der Berechnung der Spaltenstatistiken für diese Tabelle.

- `Retention` – Zahl (Ganzzahl), nicht mehr als Keine.

  Aufbewahrungsdauer für diese Tabelle.

- `StorageDescriptor` – Ein StorageDescriptor-Objekt.

  Eine Speicherbeschreibung, die Informationen zur physischen Speicherung dieser Tabelle enthält.

- `PartitionKeys` – Ein Array mit Spalte-Objekten.

  Eine Liste der Spalten, nach denen die Tabelle partitioniert ist. Es werden nur primitive Typen als Partitionsschlüssel unterstützt.

  Wenn Sie eine Tabelle erstellen, die von Amazon Athena verwendet wird, und keine `partitionKeys` angeben, müssen Sie mindestens den Wert der `partitionKeys` auf eine leere Liste festlegen. Zum Beispiel:

```
"PartitionKeys": []
```

- `ViewOriginalText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Für Apache-Hive-Kompatibilität enthalten. Wird im normalen AWS Glue Betriebsablauf nicht verwendet. Wenn es sich bei der Tabelle um eine bestimmte Athena Konfiguration handelt`VIRTUAL_VIEW`, die in Base64 codiert ist.

- `ViewExpandedText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Für Apache-Hive-Kompatibilität enthalten. Wird im normalen Betriebsablauf nicht verwendet. AWS Glue

- `TableType` – UTF-8-Zeichenfolge, nicht mehr als 255 Bytes lang.

  Der Typ dieser Tabelle. AWS Glue erstellt Tabellen mit dem `EXTERNAL_TABLE` Typ. Andere Dienste, wie z. B. Athena, können Tabellen mit zusätzlichen Tabellentypen erstellen.

  AWS Glue verwandte Tabellentypen:

  EXTERNAL_TABLE

  Hive-kompatibles Attribut – zeigt eine nicht von Hive verwaltete Tabelle an.

  GOVERNED

  Wird verwendet von AWS Lake Formation. Der AWS Glue Datenkatalog versteht es`GOVERNED`.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren die der Tabelle zugeordneten Eigenschaften.

- `TargetTable` – Ein TableIdentifier-Objekt.

  Eine `TableIdentifier`-Struktur, die eine Zieltabelle für die Verknüpfung von Ressourcen beschreibt.

- `ViewDefinition` – Ein ViewDefinitionInput-Objekt.

  Eine Struktur, die alle Informationen enthält, die die Ansicht definieren, einschließlich des Dialekts oder der Dialekte für die Ansicht und der Abfrage.

FederatedTable Struktur

Eine Datenbank, die auf eine Entität außerhalb von AWS Glue Data Catalog verweist.

Felder

- `Identifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Verbundtabelle.
- `DatabaseIdentifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Verbunddatenbank.
- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung zum externen Metaspeicher.
- `ConnectionType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Verbindungstyp, der für den Zugriff auf die Verbundtabelle verwendet wird, wobei das Protokoll oder die Methode für die Verbindung mit der externen Datenquelle angegeben wird.

Spaltenstruktur

Eine Spalte in einer `Table`.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `Column`.
- `Type` – UTF-8-Zeichenfolge, nicht mehr als 131 072 Bytes lang, passend zum Single-line string pattern.

  Der Datentyp von `Column`.

- `Comment` – Kommentar-Zeichenfolge, nicht mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Freitextkommentar.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren die der Spalte zugeordneten Eigenschaften.

StorageDescriptor Struktur

Beschreibt den physischen Speicher von Tabellendaten.

Felder

- `Columns` – Ein Array mit Spalte-Objekten.

  Eine Liste der `Columns` in der Tabelle.

- `Location` – Standort-Zeichenfolge, nicht mehr als 2056 Bytes lang, passend zum URI address multi-line string pattern.

  Der physische Speicherort der Tabelle. Standardmäßig ist dies der Lager-Speicherort, gefolgt vom Datenbank-Standort in der Lagerorganisation, gefolgt vom Namen der Tabelle.

- `AdditionalLocations` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Speicherorten, die auf den Pfad verweisen, in dem sich eine Delta-Tabelle befindet.

- `InputFormat` – Format-Zeichenfolge, nicht mehr als 128 Bytes lang, passend zum Single-line string pattern.

  Eingabeformat `SequenceFileInputFormat` (binär), `TextInputFormat` oder ein benutzerdefiniertes Format.

- `OutputFormat` – Format-Zeichenfolge, nicht mehr als 128 Bytes lang, passend zum Single-line string pattern.

Ausgabeformat `SequenceFileOutputFormat` (binär), `IgnoreKeyTextOutputFormat` oder ein benutzerdefiniertes Format.

- `Compressed` – Boolesch.

  `True`, wenn die Daten in der Tabelle komprimiert sind und `False`, wenn dies nicht der Fall ist.

- `NumberOfBuckets` – Zahl (Ganzzahl).

  Muss angegeben werden, wenn die Tabelle Dimensionsspalten enthält.

- `SerdeInfo` – Ein SerDeInfo-Objekt.

  Die serialization/deserialization (SerDe) Information.

- `BucketColumns` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste mit Reducer-Gruppierungsspalten, Cluster-Spalten und Bucketing-Spalten in der Tabelle.

- `SortColumns` – Ein Array mit Order-Objekten.

  Eine Liste mit der Sortierreihenfolge der einzelnen Buckets in der Tabelle.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Vom Benutzer angegebene Eigenschaften im Schlüssel-Wert-Format.

- `SkewedInfo` – Ein SkewedInfo-Objekt.

  Informationen über Werte, die sehr häufig in einer Spalte vorkommen (verzerrte Werte).

- `StoredAsSubDirectories` – Boolesch.

  `True`, wenn die Tabellendaten in Unterverzeichnissen gespeichert werden, andernfalls `False`.

- `SchemaReference` – Ein SchemaReference-Objekt.

  Ein Objekt, das auf ein in der Schemaregistrierung gespeichertes AWS Glue Schema verweist.

  Beim Erstellen einer Tabelle können Sie eine leere Liste von Spalten für das Schema übergeben und stattdessen eine Schemaverweisung verwenden.

SchemaReference Struktur

Ein Objekt, das auf ein in der Schemaregistry gespeichertes AWS Glue Schema verweist.

Felder

- `SchemaId` – Ein SchemaId-Objekt.

  Eine Struktur, die Schema-Identitätsfelder enthält. Entweder dies oder `SchemaVersionId` muss zur Verfügung gestellt werden.
- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige ID, die einer Version des Schemas zugewiesen ist. Entweder dies oder `SchemaId` muss zur Verfügung gestellt werden.
- `SchemaVersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

SerDeInfo Struktur

Informationen über ein serialization/deserialization Programm (SerDe), das als Extraktor und Loader dient.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des SerDe.
- `SerializationLibrary` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Normalerweise die Klasse, die das implementiert SerDe. Ein Beispiel ist `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.
- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

Diese Schlüssel-Wert-Paare definieren Initialisierungsparameter für. SerDe

Order-Struktur

Gibt die Sortierreihenfolge einer sortierten Spalte an.

Felder

- `Column` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Spalte.
- `SortOrder` – Erforderlich: Zahl (Ganzzahl), nicht mehr als 1.

  Gibt an, dass die Spalte in aufsteigender Reihenfolge (== 1) oder in absteigender Reihenfolge (==0) sortiert wird.

SkewedInfo Struktur

Gibt verzerrte Werte in einer Tabelle an. Verzerrte Werte sind solche, die mit sehr hoher Häufigkeit auftreten.

Felder

- `SkewedColumnNames` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Namen von Spalten, die verzerrte Werte enthalten.
- `SkewedColumnValues` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Werte, die so häufig auftreten, dass sie als verzerrt betrachtet werden.
- `SkewedColumnValueLocationMaps` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein Mapping verzerrter Werte zu den Spalten, die sie enthalten.

TableVersion Struktur

Gibt eine Version einer Tabelle an.

Felder

- `Table` – Ein Tabelle-Objekt.

  Die betreffende Tabelle.
- `VersionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der ID-Wert für die Identifikation dieser Tabellenversion. Eine `VersionId` ist eine Zeichenfolgendarstellung einer Ganzzahl. Jede Version wird um 1 erhöht.

TableError Struktur

Ein Fehlerdatensatz für Tabellenoperationen.

Felder

- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität muss dieser vollständig aus Kleinbuchstaben bestehen.
- `ErrorDetail` – Ein ErrorDetail-Objekt.

  Die Details zu dem Fehler.

TableVersionError Struktur

Ein Fehlerdatensatz für Tabellenversionsoperationen.

Felder

- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der betreffenden Tabelle.

- `VersionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der ID-Wert der betreffenden Version. Eine `VersionID` ist eine Zeichenfolgendarstellung einer Ganzzahl. Jede Version wird um 1 erhöht.

- `ErrorDetail` – Ein ErrorDetail-Objekt.

  Die Details zu dem Fehler.

SortCriterion Struktur

Gibt das Feld an, nach dem sortiert werden soll, und eine Sortierreihenfolge.

Felder

- `FieldName`— Wertezeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Der Name des Felds, nach dem sortiert werden soll.

- `Sort` – UTF-8-Zeichenfolge (zulässige Werte: `ASC="ASCENDING"` | `DESC="DESCENDING"`).

  Eine auf- oder absteigende Sortierung.

TableIdentifier Struktur

Eine -Struktur, die eine Zieltabelle für die Verknüpfung von Ressourcen beschreibt.

Felder

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Tabelle gespeichert ist.

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, die die Zieltabelle enthält

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Zieltabelle.

- `Region` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Region der Zieltabelle.

KeySchemaElement Struktur

Ein Partitionsschlüsselpaar bestehend aus einem Namen und einem Typ.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines Partitionsschlüssels.
- `Type` – Erforderlich: UTF-8-Zeichenfolge, nicht mehr als 131 072 Bytes lang, passend zum Single-line string pattern.

  Der Typ eines Partitionsschlüssels.

PartitionIndex Struktur

Eine Struktur für einen Partitionsindex.

Felder

- `Keys` – Erforderlich:Ein Array mit UTF-8-Zeichenfolgen, mindestens 1 Zeichenfolge.

  Die Schlüssel für den Partitionsindex.
- `IndexName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Partitionsindex.

PartitionIndexDescriptor Struktur

Ein Deskriptor für einen Partitionsindex in einer Tabelle.

Felder

- `IndexName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Partitionsindex.

- `Keys` – Erforderlich: Ein Array von KeySchemaElement-Objekten, mindestens 1 Struktur.

  Eine Liste mit einem oder mehreren Schlüsseln, wie z. B. `KeySchemaElement`-Strukturen für den Partitionsindex.

- `IndexStatus` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `DELETING` | `FAILED`).

  Der Status des Partitionsindex.

  Die folgenden Status sind möglich:

  - CREATING: Der Index wird erstellt. Wenn sich ein Index im CREATING-Zustand befindet, kann der Index oder seine Tabelle nicht gelöscht werden.
  - ACTIVE: Die Indexerstellung ist erfolgreich.
  - FAILED: Die Indexerstellung schlägt fehl.
  - DELETING: Der Index wird aus der Liste der Indizes gelöscht.

- `BackfillErrors` – Ein Array mit BackfillError-Objekten.

  Eine Liste von Fehlern, die beim Registrieren von Partitionsindizes für eine vorhandene Tabelle auftreten können.

BackfillError Struktur

Eine Liste von Fehlern, die beim Registrieren von Partitionsindizes für eine vorhandene Tabelle auftreten können.

Diese Fehler geben die Details zu den Gründen an, weshalb eine Indexregistrierung fehlgeschlagen ist, und stellen eine begrenzte Anzahl von Partitionen in der Antwort bereit, sodass Sie die fehlerhaften Partitionen beheben und die erneute Registrierung des Index probieren können. Die häufigsten Fehler, die auftreten können, lassen sich wie folgt kategorisieren:

- EncryptedPartitionError: Die Partitionen sind verschlüsselt.

- InvalidPartitionTypeDataError: Der Partitionswert entspricht nicht dem Datentyp für diese Partitionsspalte.
- MissingPartitionValueError: Die Partitionen sind verschlüsselt.
- UnsupportedPartitionCharacterError: Zeichen innerhalb des Partitionswerts werden nicht unterstützt. Beispiel: U+0000, U+0001, U+0002.
- InternalError: Jeder Fehler, der nicht zu anderen Fehlercodes gehört.

Felder

- `Code` – UTF-8-Zeichenfolge (zulässige Werte: `ENCRYPTED_PARTITION_ERROR` | `INTERNAL_ERROR` | `INVALID_PARTITION_TYPE_DATA_ERROR` | `MISSING_PARTITION_VALUE_ERROR` | `UNSUPPORTED_PARTITION_CHARACTER_ERROR`).

  Der Fehlercode für einen Fehler, der beim Registrieren von Partitionsindizes für eine vorhandene Tabelle aufgetreten ist.
- `Partitions` – Ein Array mit PartitionValueList-Objekten.

  Eine Liste einer begrenzten Anzahl an Partitionen in der Antwort.

IcebergInput Struktur

Eine Struktur, die eine im Katalog zu erstellende Apache-Iceberg-Metadatentabelle definiert.

Felder

- `MetadataOperation` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATE`).

  Ein erforderlicher Metadatenvorgang. Kann nur auf `CREATE` festgelegt werden.
- `Version` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die TabelOpenTableFormatInputIenversion für die Iceberg-Tabelle. Standard-Einstellung: 2.
- `CreateIcebergTableInput` – Ein CreateIcebergTableInput-Objekt.

  Die Konfigurationsparameter, die zum Erstellen einer neuen Iceberg-Tabelle im AWS Glue Datenkatalog erforderlich sind, einschließlich Tabelleneigenschaften und Metadatenspezifikationen.

OpenTableFormatInput Struktur

Eine Struktur, die eine Tabelle im offenen Format darstellt.

Felder

- `IcebergInput` – Ein IcebergInput-Objekt.

  Spezifiziert eine `IcebergInput`-Struktur, die eine Apache-Iceberg-Metadatentabelle definiert.

ViewDefinition Struktur

Eine Struktur, die Details für Repräsentationen enthält.

Felder

- `IsProtected` – Boolesch.

  Sie können dieses Kennzeichen auf true setzen, um die Engine anzuweisen, während der Abfrageplanung keine vom Benutzer bereitgestellten Operationen in den logischen Plan der Ansicht zu übertragen. Das Setzen dieses Flags garantiert jedoch nicht, dass das Modul die Anforderungen erfüllt. Schlagen Sie in der Dokumentation des Motors nach, welche Garantien gegebenenfalls gegeben werden.
- `Definer` – UTF-8-Zeichenfolge, nicht weniger als 20 oder mehr als 2 048 Bytes lang, passend zum Single-line string pattern.

  Der Definer einer Ansicht in SQL.
- `SubObjects` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 10 Zeichenfolgen.

  Eine Liste der Tabellen Amazon Resource Names (ARNs).
- `Representations` – Ein Array mit ViewRepresentation-Objekten, nicht weniger als 1 und nicht mehr als 1 000 Strukturen.

  Eine Liste von Repräsentationen.

ViewDefinitionInput Struktur

Eine Struktur, die Details zum Erstellen oder Aktualisieren einer AWS Glue Ansicht enthält.

Felder

- `IsProtected` – Boolesch.

  Sie können dieses Flag auf true setzen, um die Engine anzuweisen, bei der Abfrageplanung keine vom Benutzer bereitgestellten Operationen in den logischen Plan der Ansicht zu übertragen. Das Setzen dieses Flags garantiert jedoch nicht, dass das Modul die Anforderungen erfüllt. Schlagen Sie in der Dokumentation des Motors nach, welche Garantien gegebenenfalls gegeben werden.

- `Definer` – UTF-8-Zeichenfolge, nicht weniger als 20 oder mehr als 2 048 Bytes lang, passend zum Single-line string pattern.

  Der Definer einer Ansicht in SQL.

- `Representations`— Eine Reihe von ViewRepresentationInput Objekten, nicht weniger als 1 oder mehr als 10 Strukturen.

  Eine Liste von Strukturen, die den Dialekt der Ansicht und die Abfrage, die die Ansicht definiert, enthält.

- `SubObjects` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 10 Zeichenfolgen.

  Eine Liste der Basistabellen ARNs , aus denen die Ansicht besteht.

ViewRepresentation Struktur

Eine Struktur, die den Dialekt der Ansicht und die Abfrage, die die Ansicht definiert, enthält.

Felder

- `Dialect` – UTF-8-Zeichenfolge (zulässige Werte: `REDSHIFT` | `ATHENA` | `SPARK`).

  Der Dialekt der Abfrage-Engine.

- `DialectVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang.

  Die Version des Dialekts der Abfrage-Engine. Zum Beispiel 3.0.0.

- `ViewOriginalText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Die `SELECT` Anfrage, die der Kunde während `CREATE VIEW DDL` gestellt hat. Dieses SQL wird bei einer Abfrage in einer Ansicht nicht verwendet (`ViewExpandedText`wird stattdessen verwendet). `ViewOriginalText`wird beispielsweise für Fälle verwendet`SHOW CREATE VIEW`,

in denen Benutzer den ursprünglichen DDL-Befehl sehen möchten, mit dem die Ansicht erstellt wurde.

- `ViewExpandedText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Das erweiterte SQL für die Ansicht. Dieses SQL wird von Engines bei der Verarbeitung einer Abfrage in einer Ansicht verwendet. Module können während der Erstellung der Ansicht Operationen ausführen, in die sie umgewandelt `ViewOriginalText` werden sollen`ViewExpandedText`. Zum Beispiel:

  - Vollständig qualifizierte Identifikatoren: `SELECT * from table1 -> SELECT * from db1.table1`
- `ValidationConnection` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung, die zur Validierung der spezifischen Darstellung der Ansicht verwendet werden soll.
- `IsStale` – Boolesch.

  Als veraltet markierte Dialekte sind nicht mehr gültig und müssen aktualisiert werden, bevor sie in ihren jeweiligen Abfrage-Engines abgefragt werden können.

ViewRepresentationInput Struktur

Eine Struktur, die Details einer Darstellung enthält, um eine Lake Formation Formation-Ansicht zu aktualisieren oder zu erstellen.

Felder

- `Dialect` – UTF-8-Zeichenfolge (zulässige Werte: `REDSHIFT` | `ATHENA` | `SPARK`).

  Ein Parameter, der den Engine-Typ einer bestimmten Darstellung angibt.
- `DialectVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang.

  Ein Parameter, der die Version der Engine einer bestimmten Darstellung angibt.
- `ViewOriginalText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Eine Zeichenfolge, die die ursprüngliche SQL-Abfrage darstellt, die die Ansicht beschreibt.
- `ValidationConnection` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Verbindung, die verwendet werden soll, um die spezifische Darstellung der Ansicht zu überprüfen.

- `ViewExpandedText` – UTF-8-Zeichenfolge, nicht mehr als 409 600 Bytes lang.

  Eine Zeichenfolge, die die SQL-Abfrage darstellt, die die Ansicht mit erweiterter Ressource beschreibt ARNs

UpdateOpenTableFormatInput Struktur

Eingabeparameter für die Aktualisierung von Tabellen im offenen Tabellenformat in AWS Glue Data Catalog, die als Wrapper für formatspezifische Aktualisierungsvorgänge wie Apache Iceberg dienen.

Felder

- `UpdateIcebergInput` – Ein UpdateIcebergInput-Objekt.

  Apache Iceberg-spezifische Aktualisierungsparameter, die die anzuwendenden Tabellenänderungen definieren, einschließlich Schemaänderungen, Partitionsspezifikationen und Tabelleneigenschaften.

UpdateIcebergInput Struktur

Spezifische Eingabeparameter für die Aktualisierung von Apache Iceberg-Tabellen im AWS Glue Datenkatalog, die die Aktualisierungsvorgänge enthalten, die auf eine bestehende Iceberg-Tabelle angewendet werden sollen.

Felder

- `UpdateIcebergTableInput` – Erforderlich: Ein UpdateIcebergTableInput-Objekt.

  Die spezifischen Aktualisierungsvorgänge, die auf die Iceberg-Tabelle angewendet werden sollen. Sie enthalten eine Liste von Aktualisierungen, die den neuen Status der Tabelle definieren, einschließlich Schema, Partitionen und Eigenschaften.

CreateIcebergTableInput Struktur

Die Konfigurationsparameter, die zum Erstellen einer neuen Iceberg-Tabelle im AWS Glue Datenkatalog erforderlich sind, einschließlich Tabelleneigenschaften und Metadatenspezifikationen.

Felder

- `Location`— Erforderlich: Standortzeichenfolge, nicht mehr als 2056 Byte lang, entspricht. URI address multi-line string pattern

  Der S3-Speicherort, an dem die Daten der Iceberg-Tabelle gespeichert werden.

- `Schema` – Erforderlich: Ein IcebergSchema-Objekt.

  Die Schemadefinition, die die Struktur, die Feldtypen und die Metadaten für die Iceberg-Tabelle spezifiziert.

- `PartitionSpec` – Ein IcebergPartitionSpec-Objekt.

  Die Partitionierungsspezifikation, die definiert, wie die Daten der Iceberg-Tabelle organisiert und partitioniert werden, um eine optimale Abfrageleistung zu erzielen.

- `WriteOrder` – Ein IcebergSortOrder-Objekt.

  Die Spezifikation für die Sortierreihenfolge, die definiert, wie Daten innerhalb der einzelnen Partitionen angeordnet werden sollen, um die Abfrageleistung zu optimieren.

- `Properties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Schlüssel-Wert-Paare zusätzlicher Tabelleneigenschaften und Konfigurationseinstellungen für die Iceberg-Tabelle.

UpdateIcebergTableInput Struktur

Enthält die Aktualisierungsvorgänge, die auf eine bestehende Iceberg-Tabelle im AWS Glue Datenkatalog angewendet werden sollen, und definiert den neuen Status der Tabellenmetadaten.

Felder

- `Updates` – Erforderlich: Ein Array mit IcebergTableUpdate-Objekten.

  Die Liste der Tabellenaktualisierungsvorgänge, die die Änderungen spezifizieren, die an der Iceberg-Tabelle vorgenommen werden sollen, einschließlich Schemaänderungen, Partitionsspezifikationen und Tabelleneigenschaften.

IcebergSortOrder Struktur

Definiert die Spezifikation der Sortierreihenfolge für eine Iceberg-Tabelle und legt fest, wie Daten innerhalb von Partitionen angeordnet werden sollen, um die Abfrageleistung zu optimieren.

Felder

- `OrderId` – Erforderlich: Zahl (Ganzzahl).

  Der eindeutige Bezeichner für diese Sortierreihenfolgespezifikation in den Metadaten der Iceberg-Tabelle.

- `Fields` – Erforderlich: Ein Array mit IcebergSortField-Objekten.

  Die Liste der Felder und ihrer Sortierrichtungen, die die Sortierkriterien für die Daten der Iceberg-Tabelle definieren.

IcebergSortField Struktur

Definiert ein einzelnes Feld innerhalb einer Iceberg-Sortierreihenfolgenspezifikation, einschließlich des Quellfeldes, der Transformation, der Sortierrichtung und der Nullwertreihenfolge.

Felder

- `SourceId` – Erforderlich: Zahl (Ganzzahl).

  Der Bezeichner des Quellfeldes aus dem Tabellenschema, auf dem dieses Sortierfeld basiert.

- `Transform` – Erforderlich: UTF-8-Zeichenfolge.

  Die Transformationsfunktion, die vor dem Sortieren auf das Quellfeld angewendet wurde, z. B. Identity, Bucket oder Truncate.

- `Direction` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `asc="ASC"` | `desc="DESC"`).

  Die Sortierrichtung für dieses Feld, entweder aufsteigend oder absteigend.

- `NullOrder` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `nulls-first="NULLS_FIRST"` | `nulls-last="NULLS_LAST"`).

  Das Sortierverhalten für Nullwerte in diesem Feld, das angibt, ob Nullen in der Sortierreihenfolge an erster oder letzter Stelle stehen sollen.

IcebergPartitionSpec Struktur

Definiert die Partitionierungsspezifikation für eine Iceberg-Tabelle und legt fest, wie Tabellendaten organisiert und partitioniert werden, um eine optimale Abfrageleistung zu erzielen.

Felder

- `Fields` – Erforderlich: Ein Array mit IcebergPartitionField-Objekten.

  Die Liste der Partitionsfelder, die definieren, wie die Tabellendaten partitioniert werden sollen, einschließlich Quellfeldern und deren Transformationen.

- `SpecId` – Zahl (Ganzzahl).

  Die eindeutige Kennung für diese Partitionsspezifikation innerhalb der Metadatenhistorie der Iceberg-Tabelle.

IcebergPartitionField Struktur

Definiert ein einzelnes Partitionsfeld innerhalb einer Iceberg-Partitionsspezifikation, einschließlich des Quellfeldes, der Transformationsfunktion, des Partitionsnamens und der eindeutigen Kennung.

Felder

- `SourceId` – Erforderlich: Zahl (Ganzzahl).

  Der Bezeichner des Quellfeldes aus dem Tabellenschema, auf dem dieses Partitionsfeld basiert.

- `Transform` – Erforderlich: UTF-8-Zeichenfolge.

  Die Transformationsfunktion, die auf das Quellfeld angewendet wurde, um die Partition zu erstellen, z. B. Identität, Bucket, Truncate, Jahr, Monat, Tag oder Stunde.

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Single-line string pattern.

  Der Name des Partitionsfeldes, wie er in der partitionierten Tabellenstruktur erscheinen wird.

- `FieldId` – Zahl (Ganzzahl).

  Die eindeutige Kennung, die diesem Partitionsfeld in der Partitionsspezifikation der Iceberg-Tabelle zugewiesen wurde.

IcebergSchema Struktur

Definiert die Schemastruktur für eine Iceberg-Tabelle, einschließlich Felddefinitionen, Datentypen und Schema-Metadaten.

Felder

- `SchemaId` – Zahl (Ganzzahl).

  Der eindeutige Bezeichner für diese Schemaversion innerhalb der Schemaentwicklungsgeschichte der Iceberg-Tabelle.
- `IdentifierFieldIds`— Ein Array von 32-Bit-Ganzzahlen mit Vorzeichen.

  Die Liste der Feldbezeichner, die Datensätze in der Tabelle eindeutig identifizieren und für Operationen auf Zeilenebene und Deduplizierung verwendet werden.
- `Type` – UTF-8-Zeichenfolge (zulässige Werte: `struct="STRUCT"`).

  Der Stammtyp der Schemastruktur, typischerweise „struct“ für Iceberg-Tabellenschemas.
- `Fields` – Erforderlich: Ein Array mit IcebergStructField-Objekten.

  Die Liste der Felddefinitionen, aus denen das Tabellenschema besteht, einschließlich Feldnamen, Typen und Metadaten.

IcebergStructField Struktur

Definiert ein einzelnes Feld innerhalb eines Iceberg-Tabellenschemas, einschließlich seines Bezeichners, Namens, Datentyps, NULL-Zulässigkeit und Dokumentation.

Felder

- `Id` – Erforderlich: Zahl (Ganzzahl).

  Die eindeutige Kennung, die diesem Feld innerhalb des Iceberg-Tabellenschemas zugewiesen wurde und für die Schemaentwicklung und Feldverfolgung verwendet wird.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Single-line string pattern.

  Der Name des Felds, wie er im Tabellenschema und in den Abfrageoperationen erscheint.
- `Type`— Erforderlich: Eine leere Struktur mit dem Namen. `IcebergDocument`

Die Datentypdefinition für dieses Feld, die die Struktur und das Format der darin enthaltenen Daten spezifiziert.

- `Required` – Erforderlich: Boolean.

  Gibt an, ob dieses Feld im Tabellenschema erforderlich (nicht nullwertfähig) oder optional (nullwertfähig) ist.

- `Doc` – Kommentar-Zeichenfolge, nicht mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Optionaler Dokumentations- oder Beschreibungstext, der zusätzlichen Kontext zum Zweck und zur Verwendung dieses Felds bietet.

IcebergTableUpdate Struktur

Definiert einen vollständigen Satz von Aktualisierungen, die auf eine Iceberg-Tabelle angewendet werden sollen, einschließlich Schemaänderungen, Partitionierungsänderungen, Anpassungen der Sortierreihenfolge, Standortaktualisierungen und Eigenschaftsänderungen.

Felder

- `Schema` – Erforderlich: Ein IcebergSchema-Objekt.

  Die aktualisierte Schemadefinition für die Iceberg-Tabelle, in der alle Änderungen an der Feldstruktur, den Datentypen oder den Schema-Metadaten angegeben sind.

- `PartitionSpec` – Ein IcebergPartitionSpec-Objekt.

  Die aktualisierte Partitionierungsspezifikation, die definiert, wie die Tabellendaten reorganisiert und partitioniert werden sollen.

- `SortOrder` – Ein IcebergSortOrder-Objekt.

  Die aktualisierte Spezifikation für die Sortierreihenfolge, die definiert, wie Daten innerhalb von Partitionen angeordnet werden sollten, um eine optimale Abfrageleistung zu erzielen.

- `Location`— Erforderlich: Standortzeichenfolge, nicht mehr als 2056 Byte lang, entspricht. URI address multi-line string pattern

  Der aktualisierte S3-Speicherort, an dem die Daten der Iceberg-Tabelle gespeichert werden.

- `Properties` – Ein Map-Array von Schlüssel-Wert-Paaren.

Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

Jeder Wert ist eine UTF-8-Zeichenfolge.

Die Schlüssel-Wert-Paare von Tabelleneigenschaften und Konfigurationseinstellungen für die Iceberg-Tabelle wurden aktualisiert.

Operationen

- CreateTable Aktion (Python: create_table)
- UpdateTable Aktion (Python: update_table)
- DeleteTable Aktion (Python: delete_table)
- BatchDeleteTable Aktion (Python: batch_delete_table)
- GetTable Aktion (Python: get_table)
- GetTables Aktion (Python: get_tables)
- GetTableVersion Aktion (Python: get_table_version)
- GetTableVersions Aktion (Python: get_table_versions)
- DeleteTableVersion Aktion (Python: delete_table_version)
- BatchDeleteTableVersion Aktion (Python: batch_delete_table_version)
- SearchTables Aktion (Python: search_tables)
- GetPartitionIndexes Aktion (Python: get_partition_indexes)
- CreatePartitionIndex Aktion (Python: create_partition_index)
- DeletePartitionIndex Aktion (Python: delete_partition_index)
- GetColumnStatisticsForTable Aktion (Python: get_column_statistics_for_table)
- UpdateColumnStatisticsForTable Aktion (Python: update_column_statistics_for_table)
- DeleteColumnStatisticsForTable Aktion (Python: delete_column_statistics_for_table)

CreateTable Aktion (Python: create_table)

Erstellt eine neue Tabellendefinition im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die erstellt wird `Table`. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalogdatenbank, in der die neue Tabelle erstellt wird. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung für die Tabelle in der angegebenen Datenbank, die im AWS Glue Datenkatalog erstellt wird.

- `TableInput` – Erforderlich: Ein TableInput-Objekt.

  Das `TableInput`-Objekt, das die im Katalog zu erstellende Metadatentabelle definiert.

- `PartitionIndexes` – Ein Array mit PartitionIndex-Objekten, nicht mehr als 3 Strukturen.

  Eine Liste von Partitionsindizes, `PartitionIndex`-Strukturen, die in der Tabelle erstellt werden.

- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die ID der Transaktion.

- `OpenTableFormatInput` – Ein OpenTableFormatInput-Objekt.

  Gibt eine `OpenTableFormatInput`-Struktur an, wenn eine Tabelle im offenen Format erstellt wird.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

UpdateTable Aktion (Python: update_table)

Aktualisiert eine Metadatentabelle im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung für die Tabelle in der angegebenen Datenbank, die im AWS Glue Datenkatalog erstellt wird.

- `TableInput` – Ein TableInput-Objekt.

  Ein aktualisiertes `TableInput`-Objekt, das die Metadatentabelle im Katalog definiert.

- `SkipArchive` – Boolesch.

  Standardmäßig erstellt `UpdateTable` immer eine archivierte Version der Tabelle, bevor sie aktualisiert wird. Wenn `skipArchive` auf „true“ gesetzt wurde, erstellt `UpdateTable` jedoch nicht die archivierte Version.

- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Tabellinhalt aktualisiert werden soll.

- `VersionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Versions-ID, bei der der Tabelleninhalt aktualisiert werden soll.

- `ViewUpdateAction` – UTF-8-Zeichenfolge (zulässige Werte: `ADD` | `REPLACE` | `ADD_OR_REPLACE` | `DROP`).

  Der Vorgang, der beim Aktualisieren der Ansicht ausgeführt werden soll.

- `Force` – Boolesch.

  Ein Flag, das auf true gesetzt werden kann, um übereinstimmende Speicherdeskriptor- und Unterobjektübereinstimmungsanforderungen zu ignorieren.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`
- `AlreadyExistsException`

DeleteTable Aktion (Python: delete_table)

Entfernt eine Tabellendefinition aus dem Data Catalog.

> **Note**
>
> Nach Abschluss dieser Operation haben Sie keinen Zugriff mehr auf die Tabellenversionen und Partitionen, die zu der gelöschten Tabelle gehören. AWS Glue Löscht diese „verwaisten“ Ressourcen asynchron und zeitnah nach Ermessen des Services.
> Um die sofortige Löschung aller damit verbundenen Ressourcen zu gewährleisten, bevor Sie `DeleteTable` aufrufen, verwenden Sie `DeleteTableVersion` oder `BatchDeleteTableVersion` und `DeletePartition` oder `BatchDeletePartition`, um alle Ressourcen zu löschen, die zur Tabelle gehören.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Tabelle, die gelöscht werden soll. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Tabellinhalt gelöscht werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchDeleteTable Aktion (Python: batch_delete_table)

Löscht mehrere Tabellen auf einmal.

> **Note**
>
> Nach Abschluss dieser Operation haben Sie keinen Zugriff mehr auf die Tabellenversionen und Partitionen, die zu der gelöschten Tabelle gehören. AWS Glue Löscht diese „verwaisten“ Ressourcen asynchron und zeitnah nach Ermessen des Services.
> Um die sofortige Löschung aller damit verbundenen Ressourcen zu gewährleisten, bevor Sie `BatchDeleteTable` aufrufen, verwenden Sie `DeleteTableVersion` oder

`BatchDeleteTableVersion` und `DeletePartition` oder `BatchDeletePartition`, um alle Ressourcen zu löschen, die zur Tabelle gehören.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die zu löschenden Tabellen befinden. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `TablesToDelete` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der zu löschenden Tabellen.
- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Tabellinhalt gelöscht werden soll.

Antwort

- `Errors` – Ein Array mit TableError-Objekten.

  Eine Liste der Fehler, die beim Löschen der angegebenen Tabellen aufgetreten sind.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

- `GlueEncryptionException`
- `ResourceNotReadyException`

GetTable Aktion (Python: get_table)

Ruft die `Table`-Definition in einem Data Catalog für eine bestimmte Tabelle ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die die Definition abgerufen werden soll. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Tabellinhalt gelesen werden soll.
- `QueryAsOfTime` – Zeitstempel.

  Die Zeit, zu der der Tabelleninhalt gelesen werden muss. Wenn nicht festgelegt, wird die letzte Transaktions-Commit-Zeit verwendet. Kann nicht zusammen mit `TransactionId` angegeben werden.
- `IncludeStatusDetails` – Boolesch.

  Gibt an, ob Statusdetails im Zusammenhang mit einer Anforderung zur Erstellung oder Aktualisierung einer AWS Glue Datenkatalogansicht aufgenommen werden sollen.

Antwort

- `Table` – Ein Tabelle-Objekt.

  Das `Table`-Objekt, das die angegebene Tabelle definiert.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTables Aktion (Python: get_tables)

Ruft die Definitionen von einigen oder allen Tabellen in einer bestimmten `Database` ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabellen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank im Katalog, dessen Tabellen aufgelistet werden sollen. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `Expression` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum Single-line string pattern.

Ein Muster von regulären Ausdrücken. Sofern vorhanden, werden nur die Tabellen, deren Namen mit dem Muster übereinstimmen, zurückgegeben.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Gibt die maximale Anzahl der Tabellen an, die in einer Antwort zurückzugeben sind.

- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Tabellinhalt gelesen werden soll.

- `QueryAsOfTime` – Zeitstempel.

  Die Zeit, zu der der Tabelleninhalt gelesen werden muss. Wenn nicht festgelegt, wird die letzte Transaktions-Commit-Zeit verwendet. Kann nicht zusammen mit `TransactionId` angegeben werden.

- `IncludeStatusDetails` – Boolesch.

  Gibt an, ob Statusdetails im Zusammenhang mit einer Anforderung zur Erstellung oder Aktualisierung einer AWS Glue Datenkatalogansicht aufgenommen werden sollen.

- `AttributesToGet` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die vom `GetTables` Aufruf zurückgegebenen Tabellenfelder an. Dieser Parameter akzeptiert keine leere Liste. Die Anfrage muss Folgendes beinhaltenNAME.

  Im Folgenden sind die gültigen Wertekombinationen aufgeführt:

  - `NAME`- Namen aller Tabellen in der Datenbank.
  - `NAME`, `TABLE_TYPE` - Namen aller Tabellen und der Tabellentypen.

Antwort

- `TableList` – Ein Array mit Tabelle-Objekten.

  Eine Liste der angeforderten `Table`-Objekte.

- `NextToken` – UTF-8-Zeichenfolge.

Ein Fortsetzungs-Token, der vorhanden ist, wenn das aktuelle Listensegment nicht das letzte ist.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTableVersion Aktion (Python: get_table_version)

Ruft eine angegebene Version einer Tabelle ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabellen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank im Katalog, in dem sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `VersionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der ID-Wert der Tabellenversion, die abgerufen werden soll. Eine `VersionID` ist eine Zeichenfolgendarstellung einer Ganzzahl. Jede Version wird um 1 erhöht.

Antwort

- `TableVersion` – Ein TableVersion-Objekt.

  Die angeforderte Tabellenversion.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetTableVersions Aktion (Python: get_table_versions)

Ruft eine Liste der Zeichenfolgen ab, die verfügbare Versionen einer angegebenen Tabelle identifizieren.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabellen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank im Katalog, in dem sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Tabelle. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies nicht der erste Aufruf ist.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Die maximale Anzahl von Tabellenversionen, die in einer Antwort zurückzugeben sind.

Antwort

- `TableVersions` – Ein Array mit TableVersion-Objekten.

  Eine Liste der Zeichenfolgen, die verfügbare Versionen der angegebenen Tabelle identifizieren.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die Liste der verfügbaren Versionen nicht die letzte enthält.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteTableVersion Aktion (Python: delete_table_version)

Löscht eine angegebene Version einer Tabelle.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabellen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank im Katalog, in dem sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `VersionId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der ID-Wert der Tabellenversion, die gelöscht werden soll. Eine `VersionID` ist eine Zeichenfolgendarstellung einer Ganzzahl. Jede Version wird um 1 erhöht.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeleteTableVersion Aktion (Python: batch_delete_table_version)

Löscht einen angegebenen Batch von Versionen einer Tabelle.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die ID des Data Catalog, in dem sich die Tabellen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank im Katalog, in dem sich die Tabelle befindet. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle. Für Hive-Kompatibilität muss dieser Name vollständig aus Kleinbuchstaben bestehen.

- `VersionIds` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste IDs der Versionen, die gelöscht werden sollen. Eine `VersionId` ist eine Zeichenfolgendarstellung einer Ganzzahl. Jede Version wird um 1 erhöht.

Antwort

- `Errors` – Ein Array mit TableVersionError-Objekten.

  Eine Liste der Fehler, die beim Löschen der angegebenen Tabellenversionen aufgetreten sind.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

SearchTables Aktion (Python: search_tables)

Durchsucht eine Gruppe von Tabellen basierend auf Eigenschaften in den Tabellenmetadaten sowie auf der übergeordneten Datenbank. Sie können nach Text- oder Filterbedingungen suchen.

Sie können nur Tabellen abrufen, auf die Sie Zugriff haben, basierend auf den in Lake Formation definierten Sicherheitsrichtlinien. Sie benötigen mindestens einen schreibgeschützten Zugriff auf die Tabelle, damit sie zurückgegeben wird. Wenn Sie nicht auf alle Spalten in der Tabelle zugreifen können, wurden diese Spalten nicht durchsucht, wenn die Liste der Tabellen an Sie zurückgesendet wird. Wenn Sie Zugriff auf die Spalten haben, jedoch nicht auf die Daten in den Spalten, sind diese Spalten und die zugehörigen Metadaten für diese Spalten in der Suche enthalten.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung, bestehend aus *`account_id`*.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.
- `Filters` – Ein Array mit PropertyPredicate-Objekten.

  Eine Liste von Schlüssel-Wert-Paaren und ein Vergleichswert, der zum Filtern der Suchergebnisse verwendet wird. Gibt alle Entitäten zurück, die mit dem Prädikat übereinstimmen.

  Das `Comparator`-Mitglied von `PropertyPredicate` struct wird nur für Zeitfelder verwendet und kann für andere Feldtypen weggelassen werden. Auch beim Vergleichen von Zeichenfolgenwerten, z. B. `Key=Name`, wird ein Fuzzy-Match-Algorithmus verwendet. Das `Key`-Feld (z. B. der Wert des `Name`-Felds) wird auf bestimmte Satzzeichen aufgeteilt, zum Beispiel -, :, # usw. in Token. Dann ist jedes Token eine exakte Übereinstimmung im Vergleich mit dem `Value`-Mitglied von `PropertyPredicate`. Wenn zum Beispiel `Key=Name` und `Value=link` sind, werden Tabellen mit dem Namen `customer-link` und `xx-link-yy` zurückgegeben, aber `xxlinkyy` wird nicht zurückgegeben.
- `SearchText`— Wertezeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Eine Zeichenfolge, die für eine Textsuche verwendet wird.

  Wenn Sie einen Wert in Anführungszeichen angeben, werden Filter basierend auf einer exakten Übereinstimmung mit dem Wert angewendet.
- `SortCriteria` – Ein Array mit SortCriterion-Objekten, nicht mehr als 1 Struktur.

  Eine Liste von Kriterien zum Sortieren der Ergebnisse nach einem Feldnamen in auf- oder absteigender Reihenfolge.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Gibt die maximale Anzahl der Tabellen an, die in einer Antwort zurückzugeben sind.

- `ResourceShareType` – UTF-8-Zeichenfolge (zulässige Werte: `FOREIGN` | `ALL` | `FEDERATED`).

  Hier können Sie angeben, dass Sie die für Ihr Konto freigegebenen Datenbanken durchsuchen möchten. Die zulässigen Werte sind `FOREIGN` oder `ALL`.

  - Bei Auswahl von `FOREIGN` werden die Tabellen durchsucht, die für Ihr Konto freigegeben wurden.
  - Bei Auswahl von `ALL` werden die Tabellen durchsucht, die für Ihr Konto freigegeben sind, sowie die Tabellen in Ihrem lokalen Konto.

- `IncludeStatusDetails` – Boolesch.

  Gibt an, ob Statusdetails im Zusammenhang mit einer Anforderung zur Erstellung oder Aktualisierung einer AWS Glue Datenkatalogansicht aufgenommen werden sollen.

Antwort

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, der vorhanden ist, wenn das aktuelle Listensegment nicht das letzte ist.

- `TableList` – Ein Array mit Tabelle-Objekten.

  Eine Liste der angeforderten `Table`-Objekte. Die `SearchTables`-Antwort gibt nur die Tabellen zurück, auf die Sie Zugriff haben.

Fehler

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetPartitionIndexes Aktion (Python: get_partition_indexes)

Ruft die Partitionsindizes ab, die einer Tabelle zugeordnet sind.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID, in dem sich die Tabelle befindet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Datenbank an, aus der Partitionsindizes abgerufen werden sollen.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Tabelle an, für die Sie die Partitionsindizes abrufen möchten.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `PartitionIndexDescriptorList` – Ein Array mit PartitionIndexDescriptor-Objekten.

  Eine Liste von Indexdeskriptoren.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, der vorhanden ist, wenn das aktuelle Listensegment nicht das letzte ist.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ConflictException`

CreatePartitionIndex Aktion (Python: create_partition_index)

Erstellt einen angegebenen Partitionsindex in einer vorhandenen Tabelle.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID, in dem sich die Tabelle befindet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Datenbank an, in der Sie einen Partitionsindex erstellen möchten.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Tabelle an, in der Sie einen Partitionsindex erstellen möchten.
- `PartitionIndex` – Erforderlich: Ein PartitionIndex-Objekt.

  Gibt eine `PartitionIndex`-Struktur an, um einen Partitionsindex in einer vorhandenen Tabelle zu erstellen.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeletePartitionIndex Aktion (Python: delete_partition_index)

Löscht einen angegebenen Partitionsindex aus einer vorhandenen Tabelle.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID, in dem sich die Tabelle befindet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Datenbank an, aus der Sie einen Partitionsindex löschen möchten.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen einer Tabelle an, aus der Sie einen Partitionsindex löschen möchten.
- `IndexName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Partitionsindexes, der gelöscht werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ConflictException`
- `GlueEncryptionException`

GetColumnStatisticsForTable Aktion (Python: get_column_statistics_for_table)

Ruft Tabellenstatistiken von Spalten ab.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `GetTable`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `ColumnNames` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der Spaltennamen.

Antwort

- `ColumnStatisticsList` – Ein Array mit ColumnStatistics-Objekten.

  Liste von ColumnStatistics.
- `Errors` – Ein Array mit ColumnError-Objekten.

  Die Liste ColumnStatistics davon konnte nicht abgerufen werden.

Fehler

- `EntityNotFoundException`

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForTable Aktion (Python: update_column_statistics_for_table)

Erstellt oder aktualisiert Tabellenstatistiken von Spalten.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `UpdateTable`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `ColumnStatisticsList` – Erforderlich: Ein Array mit ColumnStatistics-Objekten, nicht mehr als 25 Strukturen.

  Eine Liste der Spaltenstatistiken.

Antwort

- `Errors` – Ein Array mit ColumnStatisticsError-Objekten.

  Liste von ColumnStatisticsErrors.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForTable Aktion (Python: delete_column_statistics_for_table)

Ruft Tabellenstatistiken von Spalten ab.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `DeleteTable`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `ColumnName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Spalte.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

Partitions-API

Die Partitions-API beschreibt Datentypen und Operationen für die Arbeit mit Partitionen.

Datentypen

- Partitionsstruktur
- PartitionInput Struktur
- PartitionSpecWithSharedStorageDescriptor Struktur
- PartitionListComposingSpec Struktur
- PartitionSpecProxy Struktur
- PartitionValueList Struktur
- Segmentstruktur
- PartitionError Struktur
- BatchUpdatePartitionFailureEntry Struktur
- BatchUpdatePartitionRequestEntry Struktur
- StorageDescriptor Struktur
- SchemaReference Struktur
- SerDeInfo Struktur
- SkewedInfo Struktur

Partitionsstruktur

Stellt ein Segment der Tabellendaten dar.

Felder

- `Values` – Ein UTF-8-Zeichenfolgen-Array.

  Die Werte der Partition.

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Name der Katalogdatenbank, in der die Partition angelegt wird.

- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbanktabelle, in der die Partition erstellt wird.

- `CreationTime` – Zeitstempel.

  Die Uhrzeit, zu der die Partition erstellt wurde.

- `LastAccessTime` – Zeitstempel.

  Die Uhrzeit, zu der das letzte Mal auf die Partition zugegriffen wurde.

- `StorageDescriptor` – Ein StorageDescriptor-Objekt.

  Stellt Informationen über den physischen Standort bereit, an dem die Partition gespeichert ist.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren Partitionsparameter.

- `LastAnalyzedTime` – Zeitstempel.

  Der letzte Zeitpunkt, zu dem die Spaltenstatistiken für diese Partition berechnet wurden.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalogs, in dem die Partition gespeichert ist.

PartitionInput Struktur

Die Struktur, die zum Erstellen und Aktualisieren einer Partition verwendet wird.

Felder

- `Values` – Ein UTF-8-Zeichenfolgen-Array.

  Die Werte der Partition. Obwohl dieser Parameter nicht von der SDK gefordert wird, müssen Sie diesen Parameter als einen gültigen Wert angeben.

  Die Werte für die Schlüssel für die neue Partition müssen als Array von Zeichenfolgenobjekten übergeben werden, die in derselben Reihenfolge wie die Partitionsschlüssel im Amazon S3-Präfix sortiert werden müssen. Andernfalls AWS Glue werden die Werte zu den falschen Schlüsseln hinzugefügt.
- `LastAccessTime` – Zeitstempel.

  Die Uhrzeit, zu der das letzte Mal auf die Partition zugegriffen wurde.
- `StorageDescriptor` – Ein StorageDescriptor-Objekt.

  Stellt Informationen über den physischen Standort bereit, an dem die Partition gespeichert ist.
- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren Partitionsparameter.
- `LastAnalyzedTime` – Zeitstempel.

  Der letzte Zeitpunkt, zu dem die Spaltenstatistiken für diese Partition berechnet wurden.

PartitionSpecWithSharedStorageDescriptor Struktur

Eine Partitionsspezifikation für Partitionen mit einem gemeinsamen physischen Standort.

Felder

- `StorageDescriptor` – Ein StorageDescriptor-Objekt.

Die freigegebenen physischen Speicherinformationen.

- `Partitions` – Ein Array mit Partition-Objekten.

  Eine Liste der Partitionen, die diesen physischen Standort gemeinsam verwenden.

PartitionListComposingSpec Struktur

Listet die zugehörigen Partitionen auf.

Felder

- `Partitions` – Ein Array mit Partition-Objekten.

  Eine Liste der Partitionen in der Erstellungsspezifikation.

PartitionSpecProxy Struktur

Bietet einen Stammpfad für angegebene Partitionen.

Felder

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalogdatenbank, in der die Partitionen gespeichert sind.
- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, die die Partitionen enthält.
- `RootPath` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Stammpfad des Proxys zur Adressierung der Partitionen.
- `PartitionSpecWithSharedSD` – Ein PartitionSpecWithSharedStorageDescriptor-Objekt.

  Eine Spezifikation für Partitionen, die denselben physischen Standort verwenden.
- `PartitionListComposingSpec` – Ein PartitionListComposingSpec-Objekt.

  Gibt eine Liste der Partitionen an.

PartitionValueList Struktur

Enthält eine Liste von Werten zur Definition von Partitionen.

Felder

- `Values` – Erforderlich: Ein Array von UTF-8 Zeichenketten.

  Die Liste der Werte.

Segmentstruktur

Definiert eine nicht überlappende Region der Partitionen einer Tabelle, sodass mehrere Anforderungen parallel ausgeführt werden können.

Felder

- `SegmentNumber` – Erforderlich: Zahl (Ganzzahl), nicht mehr als Keine.

  Die nullbasierte Indexnummer dieses Segments. Wenn beispielsweise die Gesamtzahl der Segmente 4 ist, liegen die `SegmentNumber`-Werte im Bereich von 0 bis 3.
- `TotalSegments` – Erforderlich:Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 10.

  Die Gesamtzahl der Segmente.

PartitionError Struktur

Enthält Informationen zu einem Partitionsfehler.

Felder

- `PartitionValues` – Ein UTF-8-Zeichenfolgen-Array.

  Die Werte, die die Partition definieren.
- `ErrorDetail` – Ein ErrorDetail-Objekt.

  Details zum Partitionsfehler.

BatchUpdatePartitionFailureEntry Struktur

Enthält Informationen zu einem Batch-Update-Partitionsfehler.

Felder

- `PartitionValueList` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der Werte zur Definition der Partitionen.
- `ErrorDetail` – Ein ErrorDetail-Objekt.

  Details zum Batch-Update-Partitionsfehler.

BatchUpdatePartitionRequestEntry Struktur

Eine Struktur, welche die Werte und die Struktur enthält, die zum Aktualisieren einer Partition verwendet werden.

Felder

- `PartitionValueList` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der Werte zur Definition der Partitionen.
- `PartitionInput` – Erforderlich: Ein PartitionInput-Objekt.

  Die Struktur, die zum Aktualisieren einer Partition verwendet wird.

StorageDescriptor Struktur

Beschreibt den physischen Speicher von Tabellendaten.

Felder

- `Columns` – Ein Array mit Spalte-Objekten.

  Eine Liste der `Columns` in der Tabelle.
- `Location` – Standort-Zeichenfolge, nicht mehr als 2056 Bytes lang, passend zum URI address multi-line string pattern.

Der physische Speicherort der Tabelle. Standardmäßig ist dies der Lager-Speicherort, gefolgt vom Datenbank-Standort in der Lagerorganisation, gefolgt vom Namen der Tabelle.

- `AdditionalLocations` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Speicherorten, die auf den Pfad verweisen, in dem sich eine Delta-Tabelle befindet.

- `InputFormat` – Format-Zeichenfolge, nicht mehr als 128 Bytes lang, passend zum Single-line string pattern.

  Eingabeformat `SequenceFileInputFormat` (binär), `TextInputFormat` oder ein benutzerdefiniertes Format.

- `OutputFormat` – Format-Zeichenfolge, nicht mehr als 128 Bytes lang, passend zum Single-line string pattern.

  Ausgabeformat `SequenceFileOutputFormat` (binär), `IgnoreKeyTextOutputFormat` oder ein benutzerdefiniertes Format.

- `Compressed` – Boolesch.

  `True`, wenn die Daten in der Tabelle komprimiert sind und `False`, wenn dies nicht der Fall ist.

- `NumberOfBuckets` – Zahl (Ganzzahl).

  Muss angegeben werden, wenn die Tabelle Dimensionsspalten enthält.

- `SerdeInfo` – Ein SerDeInfo-Objekt.

  Die Informationen zur Serialisierung/Deserialisierung (). SerDe

- `BucketColumns` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste mit Reducer-Gruppierungsspalten, Cluster-Spalten und Bucketing-Spalten in der Tabelle.

- `SortColumns` – Ein Array mit Order-Objekten.

  Eine Liste mit der Sortierreihenfolge der einzelnen Buckets in der Tabelle.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

Vom Benutzer angegebene Eigenschaften im Schlüssel-Wert-Format.

- `SkewedInfo` – Ein SkewedInfo-Objekt.

  Informationen über Werte, die sehr häufig in einer Spalte vorkommen (verzerrte Werte).

- `StoredAsSubDirectories` – Boolesch.

  `True`, wenn die Tabellendaten in Unterverzeichnissen gespeichert werden, andernfalls `False`.

- `SchemaReference` – Ein SchemaReference-Objekt.

  Ein Objekt, das auf ein in der Schemaregistry gespeichertes Schema verweist. AWS Glue

  Beim Erstellen einer Tabelle können Sie eine leere Liste von Spalten für das Schema übergeben und stattdessen eine Schemaverweisung verwenden.

SchemaReference Struktur

Ein Objekt, das auf ein in der Schemaregistry gespeichertes AWS Glue Schema verweist.

Felder

- `SchemaId` – Ein SchemaId-Objekt.

  Eine Struktur, die Schema-Identitätsfelder enthält. Entweder dies oder `SchemaVersionId` muss zur Verfügung gestellt werden.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige ID, die einer Version des Schemas zugewiesen ist. Entweder dies oder `SchemaId` muss zur Verfügung gestellt werden.

- `SchemaVersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

SerDeInfo Struktur

Informationen über ein Serialisierungs-/Deserialisierungsprogramm (SerDe), das als Extraktor und Loader dient.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  SerDeName des.

- `SerializationLibrary` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Normalerweise die Klasse, die das implementiert SerDe. Ein Beispiel ist `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.

- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine Schlüsselzeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 512 000 Bytes lang.

  Diese Schlüssel-Wert-Paare definieren Initialisierungsparameter für. SerDe

SkewedInfo Struktur

Gibt verzerrte Werte in einer Tabelle an. Verzerrte Werte sind solche, die mit sehr hoher Häufigkeit auftreten.

Felder

- `SkewedColumnNames` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Namen von Spalten, die verzerrte Werte enthalten.

- `SkewedColumnValues` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Werte, die so häufig auftreten, dass sie als verzerrt betrachtet werden.

- `SkewedColumnValueLocationMaps` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein Mapping verzerrter Werte zu den Spalten, die sie enthalten.

Operationen

- CreatePartition Aktion (Python: create_partition)
- BatchCreatePartition Aktion (Python: batch_create_partition)
- UpdatePartition Aktion (Python: update_partition)
- DeletePartition Aktion (Python: delete_partition)
- BatchDeletePartition Aktion (Python: batch_delete_partition)
- GetPartition Aktion (Python: get_partition)
- GetPartitions Aktion (Python: get_partitions)
- BatchGetPartition Aktion (Python: batch_get_partition)
- BatchUpdatePartition Aktion (Python: batch_update_partition)
- GetColumnStatisticsForPartition Aktion (Python: get_column_statistics_for_partition)
- UpdateColumnStatisticsForPartition Aktion (Python: update_column_statistics_for_partition)
- DeleteColumnStatisticsForPartition Aktion (Python: delete_column_statistics_for_partition)

CreatePartition Aktion (Python: create_partition)

Erstellt eine neue Partition.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die AWS Konto-ID des Katalogs, in dem die Partition erstellt werden soll.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatendatenbank, in der die Partition erstellt werden soll.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatendatentabelle, in der die Partition erstellt werden soll.
- `PartitionInput` – Erforderlich: Ein PartitionInput-Objekt.

  Eine `PartitionInput`-Struktur, die die zu erstellende Partition definiert.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

BatchCreatePartition Aktion (Python: batch_create_partition)

Erstellt eine oder mehrere Partitionen in einem Batchvorgang.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs, in dem die Partition erstellt wird. Derzeit sollte dies die Konto-ID sein. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatendatenbank, in der die Partition erstellt werden soll.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatendatentabelle, in der die Partition erstellt werden soll.
- `PartitionInputList` – Erforderlich: Ein Array mit PartitionInput-Objekten, nicht mehr als 100 Strukturen.

  Eine Liste der `PartitionInput`-Strukturen, die die zu erstellenden Partitionen definieren.

Antwort

- `Errors` – Ein Array mit PartitionError-Objekten.

  Die beim Erstellen der angeforderten Partitionen aufgetretenen Fehler.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdatePartition Aktion (Python: update_partition)

Aktualisiert eine Partition.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die zu aktualisierende Partition befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die entsprechende Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, in der sich die zu aktualisierende Partition befindet.
- `PartitionValueList` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

Liste von Partitionsschlüsselwerten, welche die Partition definieren.

- `PartitionInput` – Erforderlich: Ein PartitionInput-Objekt.

  Das neue Partitionsobjekt zur Aktualisierung der Partition.

  Die `Values`-Eigenschaft kann nicht geändert werden. Wenn Sie die Partitionsschlüsselwerte für eine Partition ändern möchten, löschen Sie die Partition und erstellen Sie sie neu.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeletePartition Aktion (Python: delete_partition)

Löscht eine angegebene Partition.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die zu löschende Partition befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die entsprechende Tabelle befindet.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, die die Partition enthält, die gelöscht werden soll.
- `PartitionValues` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die Werte, die die Partition definieren.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeletePartition Aktion (Python: batch_delete_partition)

Löscht eine oder mehrere Partitionen in einem Batchvorgang.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die zu löschende Partition befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die entsprechende Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, die die Partitionen enthält, die gelöscht werden sollen.

- `PartitionsToDelete` – Erforderlich: Ein Array mit PartitionValueList-Objekten, nicht mehr als 25 Strukturen.

  Eine Liste der `PartitionInput`-Strukturen, die die zu löschenden Partitionen definieren.

Antwort

- `Errors` – Ein Array mit PartitionError-Objekten.

  Die beim Löschen der angeforderten Partitionen aufgetretenen Fehler.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetPartition Aktion (Python: get_partition)

Ruft Informationen zu einer bestimmten Partition ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechende Partition befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partition befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionstabelle.
- `PartitionValues` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

Die Werte, die die Partition definieren.

Antwort

- `Partition` – Ein Partition-Objekt.

  Die angeforderten Informationen in Form eines `Partition`-Objekts.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetPartitions Aktion (Python: get_partitions)

Ruft Informationen über die Partitionen in einer Tabelle ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Partitionentabelle.

- `Expression` – Prädikatszeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Ein Ausdruck, der die Partitionen filtert, die zurückgegeben werden sollen.

  Der Ausdruck verwendet SQL-Syntax ähnlich wie die SQL-Filterklausel `WHERE`. Der Parser für SQL-Anweisungen JSQLParseranalysiert den Ausdruck.

  Operatoren: Im Folgenden finden Sie die Operatoren, die Sie im `Expression` API-Aufruf verwenden können:

  =

  Prüft, ob die Werte der beiden Operanden gleich sind. Wenn ja, wird die Bedingung wahr.

  Beispiel: Angenommen, 'variable a' hat den Wert 10, und 'variable b' hat den Wert 20.

  (a = b) ist nicht wahr.

  < >

  Prüft, ob die Werte der beiden Operanden gleich sind. Sind die Werte nicht gleich, wird die Bedingung wahr.

  Beispiel: (a < > b) ist wahr.

  >

  Prüft, ob der Wert des linken Operanden größer als der Wert des rechten Operanden ist. Wenn ja, wird die Bedingung wahr.

  Beispiel: (a > b) ist nicht wahr.

  <

  Prüft, ob der Wert des linken Operanden kleiner als der Wert des rechten Operanden ist. Wenn ja, wird die Bedingung wahr.

  Beispiel: (a < b) ist wahr.

>=

Prüft, ob der Wert des linken Operanden größer oder gleich dem Wert des rechten Operanden ist. Wenn ja, wird die Bedingung wahr.

Beispiel: (a >= b) ist nicht wahr.

<=

Prüft, ob der Wert des linken Operanden kleiner oder gleich dem Wert des rechten Operanden ist. Wenn ja, wird die Bedingung wahr.

Beispiel: (a <= b) ist wahr.

AND, OR, IN, BETWEEN, LIKE, NOT, IS NULL

Logische Operatoren.

Unterstützte Partitionsschlüsseltypen: Im Folgenden finden Sie die unterstützten Partitionsschlüssel.

- `string`
- `date`
- `timestamp`
- `int`
- `bigint`
- `long`
- `tinyint`
- `smallint`
- `decimal`

Wird ein Typ erkannt, der nicht gültig ist, wird eine Ausnahme ausgegeben.

Die folgende Liste zeigt die gültigen Operatoren für jeden Typ. Wenn Sie einen Crawler definieren, wird der `partitionKey`-Typ als `STRING`erstellt, damit er mit den Katalogpartitionen kompatibel ist.

API-Beispielaufruf:

Example

Die Tabelle `twitter_partition` hat drei Partitionen:

```
year = 2015
        year = 2016
        year = 2017
```

Example

Partition `year` gleich 2015 abrufeb

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year*=*'2015'"
```

Example

Partition `year` zwischen 2016 und 2018 (ausschließlich) abrufen

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>'2016' AND year<'2018'"
```

Example

Partition `year` zwischen 2015 und 2018 (einschließlich) abrufen. Die folgenden API-Aufrufe sind miteinander äquivalent:

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>='2015' AND year<='2018'"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year BETWEEN 2015 AND 2018"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year IN (2015,2016,2017,2018)"
```

Example

Ein Platzhalter-Partitionsfilter, wobei die Ausgabe des folgendes Aufrufs Partition Jahr = 2017 ist. Ein regulärer Ausdruck wird in `LIKE` nicht unterstützt.

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year LIKE '%7'"
```

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies nicht der erste Aufruf zum Abrufen dieser Partitionen ist.

- `Segment` – Ein Segment-Objekt.

  Das Segment der Tabellenpartitionen, das in dieser Anforderung gescannt werden soll.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Gibt die maximale Anzahl der Partitionen an, die in einer Antwort zurückzugeben sind.

- `ExcludeColumnSchema` – Boolesch.

  Wenn der Wert „true“ ist, wird das Partitionsspaltenschema nicht zurückgegeben. Nützlich, wenn Sie nur an anderen Partitionsattributen wie Partitionswerten oder Speicherort interessiert sind. Bei diesem Ansatz wird eine große Antwort vermieden, da keine doppelten Daten zurückgegeben werden.

- `TransactionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #43.

  Die Transaktions-ID, an der der Partitionsinhalt gelesen werden soll.

- `QueryAsOfTime` – Zeitstempel.

  Die Zeit, zu der der Partitionsinhalt gelesen werden muss. Wenn nicht festgelegt, wird die letzte Transaktions-Commit-Zeit verwendet. Kann nicht zusammen mit `TransactionId` angegeben werden.

Antwort

- `Partitions` – Ein Array mit Partition-Objekten.

  Eine Liste von angeforderten Partitionen.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, falls die zurückgegebene Liste der Partitionen die letzte nicht enthält.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchGetPartition Aktion (Python: batch_get_partition)

Ruft Partitionen in einer Batchanforderung ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.

- `PartitionsToGet` – Erforderlich: Ein Array mit PartitionValueList-Objekten, nicht mehr als 1000 Strukturen.

  Eine Liste der Partitionswerte, die die abzurufenden Partitionen identifizieren.

Antwort

- `Partitions` – Ein Array mit Partition-Objekten.

  Eine Liste der angeforderten Partitionen.
- `UnprocessedKeys` – Ein Array mit PartitionValueList-Objekten, nicht mehr als 1000 Strukturen.

  Eine Liste der Partitionswerte in der Anforderung, für die keine Partitionen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchUpdatePartition Aktion (Python: batch_update_partition)

Aktualisiert eine oder mehrere Partitionen in einem Batchvorgang.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs, in dem die Partition aktualisiert wird. Derzeit sollte dies die Konto-ID sein. AWS

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatendatenbank, in der die Partition aktualisiert werden soll.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Metadatentabelle, in der die Partition aktualisiert werden soll.

- `Entries` – Erforderlich: Ein Array mit BatchUpdatePartitionRequestEntry-Objekten, nicht weniger als 1 und nicht mehr als 100 Strukturen.

  Eine Liste von bis zu 100 `BatchUpdatePartitionRequestEntry`-Objekte, die aktualisiert werden sollen.

Antwort

- `Errors` – Ein Array mit BatchUpdatePartitionFailureEntry-Objekten.

  Die beim Aktualisieren der angeforderten Partitionen aufgetretenen Fehler. Eine Liste von `BatchUpdatePartitionFailureEntry`-Objekten.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

GetColumnStatisticsForPartition Aktion (Python: get_column_statistics_for_partition)

Ruft Partitionsstatistiken von Spalten ab.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `GetPartition`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `PartitionValues` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Partitionswerte, welche die Partition identifizieren.
- `ColumnNames` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der Spaltennamen.

Antwort

- `ColumnStatisticsList` – Ein Array mit ColumnStatistics-Objekten.

  Die Liste ColumnStatistics davon konnte nicht abgerufen werden.
- `Errors` – Ein Array mit ColumnError-Objekten.

  Beim Abrufen von Spaltenstatistikdaten ist ein Fehler aufgetreten.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForPartition Aktion (Python: update_column_statistics_for_partition)

Erstellt oder aktualisiert die Partitionsstatistiken von Spalten.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `UpdatePartition`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `PartitionValues` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Partitionswerte, welche die Partition identifizieren.
- `ColumnStatisticsList` – Erforderlich: Ein Array mit ColumnStatistics-Objekten, nicht mehr als 25 Strukturen.

  Eine Liste der Spaltenstatistiken.

Antwort

- `Errors` – Ein Array mit ColumnStatisticsError-Objekten.

Beim Aktualisieren von Spaltenstatistikdaten ist ein Fehler aufgetreten.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForPartition Aktion (Python: delete_column_statistics_for_partition)

Löscht die Partitionsspaltenstatistiken einer Spalte.

Die für diesen Vorgang erforderliche Identity and Access Management (IAM)-Berechtigung lautet `DeletePartition`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die entsprechenden Partitionen befinden. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Partitionen befinden.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Partitionentabelle.
- `PartitionValues` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Partitionswerte, welche die Partition identifizieren.

- `ColumnName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name der Spalte.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

Verbindungs-API

Die Connections-API beschreibt die Datentypen und die API im Zusammenhang mit der Arbeit mit Verbindungen in AWS Glue.

Themen

- Verbindungs-API
- API für Verbindungstypen
- Verbindungsmetadaten und Vorschau-API

Verbindungs-API

Die Verbindungs-API beschreibt AWS Glue Verbindungsdatentypen und die API zum Erstellen, Löschen, Aktualisieren und Auflisten von Verbindungen.

Datentypen

- Connection-Struktur
- ConnectionInput Struktur

- TestConnectionInput Struktur
- PhysicalConnectionRequirements Struktur
- GetConnectionsFilter Struktur
- AuthenticationConfiguration Struktur
- AuthenticationConfigurationInput Struktur
- OAuth2Struktur der Eigenschaften
- OAuth2PropertiesInput Struktur
- OAuth2ClientApplication Struktur
- AuthorizationCodeProperties Struktur
- BasicAuthenticationCredentials Struktur
- OAuth2Struktur der Anmeldeinformationen

Connection-Struktur

Definiert eine Verbindung zu einer Datenquelle.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindungsdefinition.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibung der Verbindung.
- `ConnectionType`— UTF-8-Zeichenfolge (gültige Werte: `JDBC` | | | `SFTP` | `MONGODB` | `KAFKA` | | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | | `GOOGLESHEETS` | `GOOGLEANALYTICS4` | `SERVICENOW` | `MARKETO` | | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `SLACK` | `STRIPE` `INTERCOM` |`SNAPCHATADS`).

  Typ der Verbindung. Derzeit wird SFTP nicht unterstützt.
- `MatchCriteria` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 10 Zeichenfolgen.

Eine Liste von Kriterien, die bei der Auswahl dieser Verbindung verwendet werden können.

- `ConnectionProperties` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (gültige Werte: `HOST` | | | `PORT` | `USERNAME="USER_NAME"` | `PASSWORD` | `ENCRYPTED_PASSWORD` | `JDBC_DRIVER_JAR_URI` | | `JDBC_DRIVER_CLASS_NAME` | `JDBC_ENGINE` | `JDBC_ENGINE_VERSION` | `CONFIG_FILES` | `INSTANCE_ID` | `JDBC_CONNECTION_URL` | `JDBC_ENFORCE_SSL` | | `CUSTOM_JDBC_CERT` | `SKIP_CUSTOM_JDBC_CERT_VALIDATION` | `CUSTOM_JDBC_CERT_STRING` | `CONNECTION_URL` | `KAFKA_BOOTSTRAP_SERVERS` | `KAFKA_SSL_ENABLED` | `KAFKA_CUSTOM_CERT` | | `KAFKA_SKIP_CUSTOM_CERT_VALIDATION` | `KAFKA_CLIENT_KEYSTORE` | `KAFKA_CLIENT_KEYSTORE_PASSWORD` | `KAFKA_CLIENT_KEY_PASSWORD` | `ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD` | `ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD` | `KAFKA_SASL_MECHANISM` | | `KAFKA_SASL_PLAIN_USERNAME` | `KAFKA_SASL_PLAIN_PASSWORD` | `ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD` | `KAFKA_SASL_SCRAM_USERNAME` | `KAFKA_SASL_SCRAM_PASSWORD` `KAFKA_SASL_SCRAM_SECRETS_ARN` `ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD` `KAFKA_SASL_GSSAPI_KEYTAB` `KAFKA_SASL_GSSAPI_KRB5_CONF` | `KAFKA_SASL_GSSAPI_SERVICE` | | `KAFKA_SASL_GSSAPI_PRINCIPAL` | `SECRET_ID` | `CONNECTOR_URL` | `CONNECTOR_TYPE` | | `CONNECTOR_CLASS_NAME` | `ENDPOINT` | `ENDPOINT_TYPE` | `ROLE_ARN` | `REGION` | `WORKGROUP_NAME` `CLUSTER_IDENTIFIER` |`DATABASE`).

  Jeder Wert ist eine Wertezeichenfolge, die nicht weniger als 1 oder mehr als 1024 Byte lang ist.

  Diese Schlüssel-Wert-Paare definieren Parameter für die Verbindung, wenn das Verbindungsschema der Version 1 verwendet wird:

  - `HOST`- Der Host-URI: entweder der vollqualifizierte Domänenname (FQDN) oder die IPv4 Adresse des Datenbank-Hosts.
  - `PORT` – Die Portnummer zwischen 1024 und 65535 des Ports, auf dem der Datenbank-Host auf Datenbankverbindungen wartet.
  - `USER_NAME` – Der Name, unter dem Sie sich bei der Datenbank anmelden. Die Wert-Zeichenfolge für `USER_NAME` ist „`USERNAME`“.
  - `PASSWORD` – Ein Passwort, falls eines verwendet wird, für den Benutzernamen.

- ENCRYPTED_PASSWORD – Wenn Sie den Verbindungspasswortschutz aktivieren, indem Sie in den Verschlüsselungseinstellungen des Data Catalog ConnectionPasswordEncryption festlegen, wird das verschlüsselte Passwort in diesem Feld gespeichert.
- JDBC_DRIVER_JAR_URI – Der Amazon Simple Storage Service (Amazon S3)-Pfad der JAR-Datei, die den zu verwendenden JDBC-treiber enthält.
- JDBC_DRIVER_CLASS_NAME – Der Klassenname des zu verwendenden JDBC-Treibers.
- JDBC_ENGINE – Der Name der zu verwendenden JDBC-Engine.
- JDBC_ENGINE_VERSION – Die zu verwendende Version der JDBC-Engine.
- CONFIG_FILES - (Zur späteren Verwendung reserviert.)
- INSTANCE_ID – Die zu verwendende Instance-ID.
- JDBC_CONNECTION_URL – Die URL für die Verbindung mit einer JDBC-Datenquelle.
- JDBC_ENFORCE_SSL- Eine boolesche Zeichenfolge (true, false) ohne Berücksichtigung von Groß- und Kleinschreibung, die angibt, ob Secure Sockets Layer (SSL) mit Hostnamenübereinstimmung für die JDBC-Verbindung auf dem Client erzwungen wird. Der Standardwert lautet „false“.
- CUSTOM_JDBC_CERT- Ein Amazon S3 S3-Standort, der das Stammzertifikat des Kunden angibt. AWS Glue verwendet dieses Stammzertifikat, um das Zertifikat des Kunden zu validieren, wenn eine Verbindung zur Kundendatenbank hergestellt wird. AWS Glue verarbeitet nur X.509-Zertifikate. Das bereitgestellte Zertifikat muss DER-codiert sein und im Base64-PEM-Codierungsformat bereitgestellt werden.
- SKIP_CUSTOM_JDBC_CERT_VALIDATION- Standardmäßig ist `false` das. AWS Glue validiert den Signaturalgorithmus und den Subject Public Key Algorithm für das Kundenzertifikat. Die einzigen zulässigen Algorithmen für den Signaturalgorithmus sind SHA256with RSA, RSA oder SHA384with RSA. SHA512with Die Schlüssellänge für den Algorithmus für den öffentlichen Schlüssel muss mindestens 2048 betragen. Sie können den Wert dieser Eigenschaft auf `true` festlegen, um die AWS Glue-Validierung des Kundenzertifikats zu überspringen.
- CUSTOM_JDBC_CERT_STRING- Eine benutzerdefinierte JDBC-Zertifikatszeichenfolge, die für den Domänenabgleich oder den Abgleich definierter Namen verwendet wird, um einen Angriff zu verhindern. man-in-the-middle In Oracle Database wird sie als SSL_SERVER_CERT_DN verwendet. In Microsoft SQL Server wird sie als `hostNameInCertificate` verwendet.
- CONNECTION_URL – Die URL für die Verbindung mit einer allgemeinen Datenquelle (nicht JDBC).
- SECRET_ID – Die geheime ID, die für den Secrets Manager der Anmeldeinformationen verwendet wird.

- `CONNECTOR_URL` – Die Connector-URL für eine MARKETPLACE- oder CUSTOM-Verbindung.
- `CONNECTOR_TYPE` – Der Konnektor-Typ für eine MARKETPLACE- oder CUSTOM-Verbindung.
- `CONNECTOR_CLASS_NAME` – Die Konnektor-Klassenname für eine MARKETPLACE- oder CUSTOM-Verbindung.
- `KAFKA_BOOTSTRAP_SERVERS` - Eine durch Kommata getrennte Liste von Host- und Portpaaren, die die Adressen der Apache Kafka Broker in einem Kafka-Cluster sind, zu dem ein Kafka-Client eine Verbindung herstellt und Bootstrapping durchführt.
- `KAFKA_SSL_ENABLED` – Ob SSL auf einer Apache-Kafka-Verbindung aktiviert oder deaktiviert werden soll. Der Standardwert ist „true“.
- `KAFKA_CUSTOM_CERT` – Die Amazon-S3-URL für die private CA-Zertifikatdatei (PEM-Format). Der Standardwert ist eine leere Zeichenfolge.
- `KAFKA_SKIP_CUSTOM_CERT_VALIDATION`— Ob die Validierung der CA-Zertifikatsdatei übersprungen werden soll oder nicht. AWS Glue validiert für drei Algorithmen: SHA256with RSA, RSA und SHA384with RSA. SHA512with Der Standardwert ist „false“.
- `KAFKA_CLIENT_KEYSTORE` – Der Amazon-S3-Speicherort der Client-Keystore-Datei für die clientseitige Kafka-Authentifizierung (optional).
- `KAFKA_CLIENT_KEYSTORE_PASSWORD` – Das Passwort für den Zugriff auf den bereitgestellten Keystore (optional).
- `KAFKA_CLIENT_KEY_PASSWORD` – Ein Keystore kann aus mehreren Schlüsseln bestehen, also ist dies das Passwort für den Zugriff auf den Clientschlüssel, der mit dem serverseitigen Kafka-Schlüssel verwendet werden soll (optional).
- `ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD`- Die verschlüsselte Version des Keystore-Passworts des Kafka-Clients (falls der Benutzer die Einstellung Passwörter verschlüsseln ausgewählt hat). AWS Glue
- `ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD`- Die verschlüsselte Version des Kafka-Client-Schlüsselkennworts (falls der Benutzer die Einstellung Passwörter AWS Glue verschlüsseln ausgewählt hat).
- `KAFKA_SASL_MECHANISM`-`"SCRAM-SHA-512"`, `"GSSAPI""AWS_MSK_IAM"`, oder. `"PLAIN"` Dies sind die unterstützten SASL-Mechanismen.
- `KAFKA_SASL_PLAIN_USERNAME`- Ein Klartext-Benutzername, der zur Authentifizierung mit dem „PLAIN“ -Mechanismus verwendet wird.
- `KAFKA_SASL_PLAIN_PASSWORD`- Ein Klartext-Passwort, das zur Authentifizierung mit dem „PLAIN“ -Mechanismus verwendet wird.

- `ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD`— Die verschlüsselte Version des Kafka SASL PLAIN-Passworts (falls der Benutzer die Einstellung Passwörter AWS Glue verschlüsseln ausgewählt hat).
- `KAFKA_SASL_SCRAM_USERNAME` – Ein Klartext-Benutzername, der zur Authentifizierung mit dem Mechanismus „SCRAM-SHA-512" verwendet wird.
- `KAFKA_SASL_SCRAM_PASSWORD` – Ein Klartext-Passwort, das zur Authentifizierung mit dem Mechanismus „SCRAM-SHA-512" verwendet wird.
- `ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD`- Die verschlüsselte Version des Kafka-SASL-SCRAM-Passworts (falls der Benutzer die Einstellung Passwörter verschlüsseln ausgewählt hat). AWS Glue
- `KAFKA_SASL_SCRAM_SECRETS_ARN`- Der Amazon-Ressourcenname eines Geheimnisses in AWS Secrets Manager.
- `KAFKA_SASL_GSSAPI_KEYTAB` – Der S3-Speicherort einer Kerberos-`keytab`-Datei. Ein Keytab speichert Langzeitschlüssel für ein oder mehrere Prinzipale. Weitere Informationen finden Sie unter MIT-Kerberos-Dokumentation: Keytab.
- `KAFKA_SASL_GSSAPI_KRB5_CONF` – Der S3-Speicherort einer Kerberos-`krb5.conf`-Datei. Eine krb5.conf speichert Kerberos-Konfigurationsinformationen, z. B. den Standort des KDC-Servers. Weitere Informationen finden Sie in der MIT-Kerberos-Dokumentation: krb5.conf.
- `KAFKA_SASL_GSSAPI_SERVICE` – Der Name des Kerberos-Services, wie er in Ihrer Kafka-Konfiguration mit `sasl.kerberos.service.name` festgelegt wurde.
- `KAFKA_SASL_GSSAPI_PRINCIPAL`- Der Name des Kerberos-Prinzials, der von verwendet wird. AWS Glue Weitere Informationen finden Sie in der Kafka-Dokumentation: Konfigurieren von Kafka-Brokern.
- `ROLE_ARN`- Die Rolle, die für die Ausführung von Abfragen verwendet werden soll.
- `REGION`- Die AWS Region, in der Abfragen ausgeführt werden.
- `WORKGROUP_NAME`— Der Name einer serverlosen Amazon Redshift Redshift-Arbeitsgruppe oder Amazon Athena Athena-Arbeitsgruppe, in der Abfragen ausgeführt werden.
- `CLUSTER_IDENTIFIER`— Die Cluster-ID eines Amazon Redshift Redshift-Clusters, in dem Abfragen ausgeführt werden.
- `DATABASE`- Die Amazon Redshift Redshift-Datenbank, mit der Sie eine Verbindung herstellen.

- `SparkProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

Spezifische Verbindungseigenschaften für die Spark-Rechenumgebung.

- `AthenaProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Spezifische Verbindungseigenschaften für die Athena-Rechenumgebung.

- `PythonProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Verbindungseigenschaften, die für die Python-Rechenumgebung spezifisch sind.

- `PhysicalConnectionRequirements` – Ein PhysicalConnectionRequirements-Objekt.

  Die physischen Verbindungsanforderungen, wie z. B. Virtual Private Cloud (VPC) und`SecurityGroup`, die erforderlich sind, um diese Verbindung erfolgreich herzustellen.

- `CreationTime` – Zeitstempel.

  Der Zeitstempel des Zeitpunkts, zu dem diese Verbindungsdefinition erstellt wurde.

- `LastUpdatedTime` – Zeitstempel.

  Der Zeitstempel der letzten Aktualisierung der Verbindungsdefinition.

- `LastUpdatedBy` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Benutzer, die Gruppe oder die Rolle, von dem bzw. der diese Verbindung zuletzt aktualisiert wurde.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `READY` | `IN_PROGRESS` | `FAILED`).

  Der Status der Verbindung. Dabei kann es sich um `READY`, `IN_PROGRESS` oder `FAILED` handeln.

- `StatusReason`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 16384 Byte lang.

Der Grund für den Verbindungsstatus.

- `LastConnectionValidationTime` – Zeitstempel.

  Ein Zeitstempel der Uhrzeit, zu der diese Verbindung zuletzt validiert wurde.

- `AuthenticationConfiguration` – Ein AuthenticationConfiguration-Objekt.

  Die Authentifizierungseigenschaften der Verbindung.

- `ConnectionSchemaVersion`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 2.

  Die Version des Verbindungsschemas für diese Verbindung. Version 2 unterstützt Eigenschaften für bestimmte Rechenumgebungen.

- `CompatibleComputeEnvironments` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Computerumgebungen, die mit der Verbindung kompatibel sind.

ConnectionInput Struktur

Eine Struktur, die zum Angeben einer Verbindung verwendet wird, die erstellt oder aktualisiert werden soll.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibung der Verbindung.

- `ConnectionType`— Erforderlich: UTF-8-String (gültige Werte: `JDBC` | `SFTP` | `MONGODB` | | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | `GOOGLEANALYTICS4` | `SERVICENOW` | | `MARKETO` | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | `HUBSPOT` | | `FACEBOOKADS` | `INSTAGRAMADS` | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | | `SLACK` `STRIPE` `INTERCOM` |`SNAPCHATADS`).

  Typ der Verbindung. Derzeit werden folgende Typen unterstützt:

- JDBC – Gibt eine Verbindung mit einer Datenbank über Java Database Connectivity (JDBC) an.

  JDBCVerbindungen verwenden Folgendes. ConnectionParameters

  - Erforderlich: Alle von (HOST, PORT, JDBC_ENGINE) oder JDBC_CONNECTION_URL.
  - Erforderlich: Alle (USERNAME, PASSWORD,) oder SECRET_ID.
  - Optional: JDBC_ENFORCE_SSL, CUSTOM_JDBC_CERT, CUSTOM_JDBC_CERT_STRING, SKIP_CUSTOM_JDBC_CERT_VALIDATION. Diese Parameter werden verwendet, um SSL mit JDBC zu konfigurieren.
- KAFKA - Bezeichnet eine Verbindung zu einer Apache-Kafka-Streaming-Plattform.

  KAFKAVerbindungen verwenden Folgendes ConnectionParameters.

  - Erforderlich: KAFKA_BOOTSTRAP_SERVERS.
  - Optional: KAFKA_SSL_ENABLED, KAFKA_CUSTOM_CERT, KAFKA_SKIP_CUSTOM_CERT_VALIDATION. Diese Parameter werden verwendet, um SSL mit KAFKA zu konfigurieren.
  - Optional: KAFKA_CLIENT_KEYSTORE, KAFKA_CLIENT_KEYSTORE_PASSWORD, KAFKA_CLIENT_KEY_PASSWORD, ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD, ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD. Diese Parameter werden verwendet, um TLS-Client-Konfiguration mit SSL in KAFKA zu konfigurieren.
  - Fakultativ:KAFKA_SASL_MECHANISM. Kann als SCRAM-SHA-512, GSSAPI oder AWS_MSK_IAM angegeben werden.
  - Optional: KAFKA_SASL_SCRAM_USERNAME, KAFKA_SASL_SCRAM_PASSWORD, ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD. Diese Parameter werden verwendet, um SASL/SCRAM-SHA-512-Authentifizierung mit KAFKA zu konfigurieren.
  - Optional: KAFKA_SASL_GSSAPI_KEYTAB, KAFKA_SASL_GSSAPI_KRB5_CONF, KAFKA_SASL_GSSAPI_SERVICE, KAFKA_SASL_GSSAPI_PRINCIPAL. Diese Parameter werden verwendet, um die SASL/GSSAPI-Authentifizierung mit KAFKA zu konfigurieren.
- MONGODB – Gibt eine Verbindung zu einer MongoDB-Dokumentendatenbank an.

  MONGODBVerbindungen verwenden Folgendes ConnectionParameters.

  - Erforderlich: CONNECTION_URL.
  - Erforderlich: Alle von (USERNAME, PASSWORD) oder SECRET_ID.
- VIEW_VALIDATION_REDSHIFT— Bezeichnet eine Verbindung, die für die Überprüfung von Ansichten durch Amazon Redshift verwendet wird.

- `VIEW_VALIDATION_ATHENA`- Bezeichnet eine Verbindung, die für die Überprüfung von Ansichten durch Amazon Athena verwendet wird.
- `NETWORK` – Bezeichnet eine Netzwerkverbindung zu einer Datenquelle in einer Amazon Virtual Private Cloud-Umgebung (Amazon VPC).

  `NETWORK`Verbindungen sind nicht erforderlich. ConnectionParameters Geben Sie stattdessen eine an PhysicalConnectionRequirements.
- `MARKETPLACE`- Verwendet Konfigurationseinstellungen, die in einem Connector enthalten sind AWS Marketplace , um aus Datenspeichern zu lesen und in diese zu schreiben, die nicht systemintern von AWS Glue unterstützt werden.

  `MARKETPLACE`Verbindungen verwenden Folgendes ConnectionParameters.
  - Erforderlich: `CONNECTOR_TYPE`, `CONNECTOR_URL`, `CONNECTOR_CLASS_NAME`, `CONNECTION_URL`.
  - Erforderlich für `JDBC` `CONNECTOR_TYPE`-Verbindungen: Alle von (`USERNAME`, `PASSWORD`) oder `SECRET_ID`.
- `CUSTOM` – Verwendet die in einem benutzerdefinierten Konnektor enthaltenen Konfigurationseinstellungen, der zum Lesen und Schreiben in nicht nativ von AWS Glue unterstützten Datastores erworben wurde.

Darüber hinaus wird ein `ConnectionType` für die folgenden SaaS-Konnektoren unterstützt:

- `FACEBOOKADS`- Bezeichnet eine Verbindung zu Facebook-Werbeanzeigen.
- `GOOGLEADS`- Bezeichnet eine Verbindung zu Google Ads.
- `GOOGLESHEETS`- Bezeichnet eine Verbindung zu Google Sheets.
- `GOOGLEANALYTICS4`- Bezeichnet eine Verbindung zu Google Analytics 4.
- `HUBSPOT`- Bezeichnet eine Verbindung zu. HubSpot
- `INSTAGRAMADS`- Bezeichnet eine Verbindung zu Instagram-Anzeigen.
- `INTERCOM`- Bezeichnet eine Verbindung zu Intercom.
- `JIRACLOUD`- Bezeichnet eine Verbindung zu Jira Cloud.
- `MARKETO`— Bezeichnet eine Verbindung zu Adobe Marketo Engage.
- `NETSUITEERP`- Bezeichnet eine Verbindung zu Oracle. NetSuite
- `SALESFORCE`— Bezeichnet eine Verbindung zu Salesforce mithilfe von Authentifizierung. OAuth
- `SALESFORCEMARKETINGCLOUD`— Bezeichnet eine Verbindung zur Salesforce Marketing Cloud.

- `SALESFORCEPARDOT`— Bezeichnet eine Verbindung zu Salesforce Marketing Cloud Account Engagement (MCAE).
- `SAPODATA`- Bezeichnet eine Verbindung zu SAP. OData
- `SERVICENOW`- Bezeichnet eine Verbindung zu. ServiceNow
- `SLACK`- Bezeichnet eine Verbindung zu Slack.
- `SNAPCHATADS`- Bezeichnet eine Verbindung zu Snapchat Ads.
- `STRIPE`- Bezeichnet eine Verbindung zu Stripe.
- `ZENDESK`- Bezeichnet eine Verbindung zu Zendesk.
- `ZOHOCRM`- Bezeichnet eine Verbindung zu Zoho CRM.

Weitere Informationen zu den Verbindungsparametern, die für einen bestimmten Connector benötigt werden, finden Sie in der Dokumentation zum Connector unter AWS Glue Verbindung hinzufügen im Benutzerhandbuch. AWS Glue

`SFTP` wird nicht unterstützt.

Weitere Informationen darüber, wie optionale Funktionen zur Konfiguration verwendet ConnectionProperties werden AWS Glue, finden Sie unter AWS Glue Verbindungseigenschaften.

Weitere Informationen darüber, wie optionale Funktionen zur Konfiguration von Funktionen in AWS Glue Studio verwendet ConnectionProperties werden, finden Sie unter Verwenden von Konnektoren und Verbindungen.

- `MatchCriteria` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 10 Zeichenfolgen.

  Eine Liste von Kriterien, die bei der Auswahl dieser Verbindung verwendet werden können.

- `ConnectionProperties` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (gültige Werte: `HOST` | | | `PORT` | `USERNAME="USER_NAME"` | `PASSWORD` | `ENCRYPTED_PASSWORD` | `JDBC_DRIVER_JAR_URI` | `JDBC_DRIVER_CLASS_NAME` | | `JDBC_ENGINE` | `JDBC_ENGINE_VERSION` | `CONFIG_FILES` | `INSTANCE_ID` | `JDBC_CONNECTION_URL` | `JDBC_ENFORCE_SSL` | `CUSTOM_JDBC_CERT` | | `SKIP_CUSTOM_JDBC_CERT_VALIDATION` | `CUSTOM_JDBC_CERT_STRING` | `CONNECTION_URL` | `KAFKA_BOOTSTRAP_SERVERS` | `KAFKA_SSL_ENABLED` | `KAFKA_CUSTOM_CERT` | `KAFKA_SKIP_CUSTOM_CERT_VALIDATION` | `KAFKA_CLIENT_KEYSTORE` | | `KAFKA_CLIENT_KEYSTORE_PASSWORD` |

KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | KAFKA_SASL_MECHANISM | KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD | | ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD | KAFKA_SASL_SCRAM_USERNAME | KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN | ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD KAFKA_SASL_GSSAPI_KEYTAB KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE | | KAFKA_SASL_GSSAPI_PRINCIPAL | SECRET_ID | CONNECTOR_URL | CONNECTOR_TYPE | | CONNECTOR_CLASS_NAME | ENDPOINT | ENDPOINT_TYPE | ROLE_ARN | REGION | WORKGROUP_NAME CLUSTER_IDENTIFIER |DATABASE).

Jeder Wert ist eine Wertezeichenfolge, die nicht weniger als 1 oder mehr als 1024 Byte lang ist.

Diese Schlüssel-Wert-Paare definieren Parameter für die Verbindung.

- `SparkProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Spezifische Verbindungseigenschaften für die Spark-Rechenumgebung.

- `AthenaProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Spezifische Verbindungseigenschaften für die Athena-Rechenumgebung.

- `PythonProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Verbindungseigenschaften, die für die Python-Rechenumgebung spezifisch sind.

- `PhysicalConnectionRequirements` – Ein PhysicalConnectionRequirements-Objekt.

Die physischen Verbindungsanforderungen, wie Virtual Private Cloud (VPC) und`SecurityGroup`, die erforderlich sind, um diese Verbindung erfolgreich herzustellen.

- `AuthenticationConfiguration` – Ein AuthenticationConfigurationInput-Objekt.

  Die Authentifizierungseigenschaften der Verbindung.

- `ValidateCredentials` – Boolesch.

  Eine Markierung zur Überprüfung der Anmeldeinformationen beim Erstellen der Verbindung. Der Standardwert ist „true“.

- `ValidateForComputeEnvironments` – Ein UTF-8-Zeichenfolgen-Array.

  Die Computerumgebungen, anhand derer die angegebenen Verbindungseigenschaften validiert werden.

TestConnectionInput Struktur

Eine Struktur, die verwendet wird, um das Testen einer Verbindung zu einem Dienst zu spezifizieren.

Felder

- `ConnectionType`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `JDBC` | | | `SFTP` | `MONGODB` | `KAFKA` | | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | `GOOGLEANALYTICS4` | | `SERVICENOW` | `MARKETO` | `SAPODATA` | `ZENDESK` | | `JIRACLOUD` | `NETSUITEERP` | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `SLACK` | `STRIPE` `INTERCOM` |`SNAPCHATADS`).

  Der Typ der zu testenden Verbindung. Dieser Vorgang ist nur für die `SALESFORCE` Verbindungstypen `JDBC` oder verfügbar.

- `ConnectionProperties` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (gültige Werte: `HOST` | | `PORT` | `USERNAME="USER_NAME"` | `PASSWORD` | `ENCRYPTED_PASSWORD` | | `JDBC_DRIVER_JAR_URI` | `JDBC_DRIVER_CLASS_NAME` | `JDBC_ENGINE` | `JDBC_ENGINE_VERSION` | `CONFIG_FILES` | `INSTANCE_ID` | `JDBC_CONNECTION_URL` | `JDBC_ENFORCE_SSL` | `CUSTOM_JDBC_CERT` | `SKIP_CUSTOM_JDBC_CERT_VALIDATION` | `CUSTOM_JDBC_CERT_STRING` |

| CONNECTION_URL | KAFKA_BOOTSTRAP_SERVERS | KAFKA_SSL_ENABLED | KAFKA_CUSTOM_CERT | KAFKA_SKIP_CUSTOM_CERT_VALIDATION | KAFKA_CLIENT_KEYSTORE | KAFKA_CLIENT_KEYSTORE_PASSWORD | KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | KAFKA_SASL_MECHANISM | | KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD | ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD | KAFKA_SASL_SCRAM_USERNAME | KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN | ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD KAFKA_SASL_GSSAPI_KEYTAB KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE | | KAFKA_SASL_GSSAPI_PRINCIPAL | SECRET_ID | CONNECTOR_URL | CONNECTOR_TYPE | | CONNECTOR_CLASS_NAME | ENDPOINT | ENDPOINT_TYPE | ROLE_ARN | REGION | WORKGROUP_NAME CLUSTER_IDENTIFIER |DATABASE).

Jeder Wert ist eine Wertezeichenfolge, die nicht weniger als 1 oder mehr als 1024 Byte lang ist.

Die Schlüssel-Wert-Paare, die Parameter für die Verbindung definieren.

JDBC-Verbindungen verwenden die folgenden Verbindungseigenschaften:

- Erforderlich: Alle von (HOST, PORT, JDBC_ENGINE) oder JDBC_CONNECTION_URL.
- Erforderlich: Alle (USERNAME, PASSWORD,) oder SECRET_ID.
- Optional: JDBC_ENFORCE_SSL, CUSTOM_JDBC_CERT, CUSTOM_JDBC_CERT_STRING, SKIP_CUSTOM_JDBC_CERT_VALIDATION. Diese Parameter werden verwendet, um SSL mit JDBC zu konfigurieren.

Für SALESFORCE-Verbindungen muss das AuthenticationConfiguration Mitglied konfiguriert werden.

- AuthenticationConfiguration – Ein AuthenticationConfigurationInput-Objekt.

  Eine Struktur, die die Authentifizierungskonfiguration in der TestConnection Anfrage enthält. Erforderlich für eine Verbindung zu Salesforce mithilfe von OAuth Authentifizierung.

PhysicalConnectionRequirements Struktur

Die OAuth Client-App als GetConnection Antwort.

Felder

- `SubnetId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Subnetz-ID, die von der Verbindung verwendet wird.
- `SecurityGroupIdList` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 50 Zeichenfolgen.

  Die Sicherheitsgruppen-ID-Liste, die von der Verbindung verwendet wird.
- `AvailabilityZone` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Availability Zone der Verbindung.

GetConnectionsFilter Struktur

Filtert die von der `GetConnections`-API-Operation zurückgegebenen Verbindungsdefinitionen.

Felder

- `MatchCriteria` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 10 Zeichenfolgen.

  Eine Kriterienzeichenfolge, die den Kriterien entsprechen muss, die in der Verbindungsdefinition hinterlegt sind, damit diese Verbindungsdefinition zurückgegeben wird.
- `ConnectionType`— UTF-8-Zeichenfolge (gültige Werte: `JDBC` | | | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | `GOOGLEANALYTICS4` | | `SERVICENOW` | `MARKETO` | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | | `SLACK` `STRIPE` `INTERCOM` |`SNAPCHATADS`).

  Der Typ von Verbindungen, der zurückgegeben werden soll. Derzeit wird SFTP nicht unterstützt.
- `ConnectionSchemaVersion`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 2.

  Gibt an, ob die Verbindung mit Schemaversion 1 oder 2 erstellt wurde.

AuthenticationConfiguration Struktur

Eine Struktur, die die Authentifizierungskonfiguration enthält.

Felder

- `AuthenticationType` – UTF-8-Zeichenfolge (zulässige Werte: `BASIC` | `OAUTH2` | `CUSTOM` | `IAM`).

  Eine Struktur, die die Authentifizierungskonfiguration enthält.

- `SecretArn` – UTF-8-Zeichenfolge, die Custom string pattern #36 entspricht.

  Der geheime Manager-ARN zum Speichern von Anmeldeinformationen.

- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

  Der Amazon-Ressourcenname (ARN) des KMS-Schlüssels, der zur Verschlüsselung vertraulicher Authentifizierungsinformationen verwendet wird. Dieser Schlüssel wird verwendet, um Anmeldeinformationen und andere sensible Daten zu schützen, die in der Authentifizierungskonfiguration gespeichert sind.

- `OAuth2Properties` – Ein OAuth2Eigenschaften-Objekt.

  Die Eigenschaften für die OAuth2 Authentifizierung.

AuthenticationConfigurationInput Struktur

Eine Struktur, die die Authentifizierungskonfiguration in der CreateConnection Anfrage enthält.

Felder

- `AuthenticationType` – UTF-8-Zeichenfolge (zulässige Werte: `BASIC` | `OAUTH2` | `CUSTOM` | `IAM`).

  Eine Struktur, die die Authentifizierungskonfiguration in der CreateConnection Anfrage enthält.

- `OAuth2Properties` – Ein OAuth2PropertiesInput-Objekt.

  Die Eigenschaften für die OAuth2 Authentifizierung in der CreateConnection Anfrage.

- `SecretArn` – UTF-8-Zeichenfolge, die Custom string pattern #36 entspricht.

  Der geheime Manager-ARN zum Speichern von Anmeldeinformationen in der CreateConnection Anfrage.

- `KmsKeyArn` – UTF-8-Zeichenfolge, die Custom string pattern #29 entspricht.

Der ARN des KMS-Schlüssels, der zum Verschlüsseln der Verbindung verwendet wurde. Es wurde nur eine als Eingabe in die Anfrage aufgenommen und im Secret Manager gespeichert.

- `BasicAuthenticationCredentials` – Ein BasicAuthenticationCredentials-Objekt.

  Die Anmeldeinformationen, die verwendet werden, wenn der Authentifizierungstyp Standardauthentifizierung ist.

- `CustomAuthenticationCredentials` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 2048 Byte lang ist.

  Die Anmeldeinformationen, die verwendet werden, wenn der Authentifizierungstyp benutzerdefinierte Authentifizierung ist.

OAuth2Struktur der Eigenschaften

Eine Struktur, die Eigenschaften für die OAuth2 Authentifizierung enthält.

Felder

- `OAuth2GrantType` – UTF-8-Zeichenfolge (zulässige Werte: `AUTHORIZATION_CODE` | `CLIENT_CREDENTIALS` | `JWT_BEARER`).

  Die Art des OAuth2 Zuschusses. Beispiel: `AUTHORIZATION_CODE`, `JWT_BEARER` oder `CLIENT_CREDENTIALS`.

- `OAuth2ClientApplication` – Ein OAuth2ClientApplication-Objekt.

  Der Typ der Client-Anwendung. Zum Beispiel AWS_MANAGED oder USER_MANAGED.

- `TokenUrl` – UTF-8-Zeichenfolge, nicht mehr als 256 Bytes lang, passend zum Custom string pattern #40.

  Die URL des Authentifizierungsservers des Anbieters, um einen Autorisierungscode gegen ein Zugriffstoken auszutauschen.

- `TokenUrlParametersMap` – Ein Map-Array von Schlüssel-Wert-Paaren.

Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 512 Byte lang ist.

Eine Zuordnung von Parametern, die der GET Token-Anfrage hinzugefügt werden.

OAuth2PropertiesInput Struktur

Eine Struktur, die Eigenschaften für OAuth2 in der CreateConnection Anfrage enthält.

Felder

- OAuth2GrantType – UTF-8-Zeichenfolge (zulässige Werte: AUTHORIZATION_CODE | CLIENT_CREDENTIALS | JWT_BEARER).

  Der OAuth2 Zuschusstyp in der CreateConnection Anfrage. Beispiel: AUTHORIZATION_CODE, JWT_BEARER oder CLIENT_CREDENTIALS.
- OAuth2ClientApplication – Ein OAuth2ClientApplication-Objekt.

  Der Typ der Client-Anwendung in der CreateConnection Anfrage. Zum Beispiel AWS_MANAGED oder USER_MANAGED.
- TokenUrl – UTF-8-Zeichenfolge, nicht mehr als 256 Bytes lang, passend zum Custom string pattern #40.

  Die URL des Authentifizierungsservers des Anbieters, um einen Autorisierungscode gegen ein Zugriffstoken auszutauschen.
- TokenUrlParametersMap – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 512 Byte lang ist.

  Eine Zuordnung von Parametern, die der GET Token-Anfrage hinzugefügt werden.
- AuthorizationCodeProperties – Ein AuthorizationCodeProperties-Objekt.

  Der Satz von Eigenschaften, die für den OAuth2 AUTHORIZATION_CODE Zuschusstyp erforderlich sind.

- `OAuth2Credentials` – Ein OAuth2Erweitern Sie im angezeigten Detailbereich die Option-Objekt.

  Die Anmeldeinformationen, die verwendet werden, wenn der Authentifizierungstyp OAuth2 Authentifizierung ist.

OAuth2ClientApplication Struktur

Die für die Verbindung verwendete OAuth2 Client-App.

Felder

- `UserManagedClientApplicationClientId` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum Custom string pattern #37.

  Die ClientID der Client-Anwendung, falls ja. ClientAppType `USER_MANAGED`
- `AWSManagedClientApplicationReference` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum Custom string pattern #37.

  Der Verweis auf die SaaS-seitige Client-App, die verwaltet wird AWS .

AuthorizationCodeProperties Struktur

Der Satz von Eigenschaften, die für den Workflow mit der Art des OAuth2 `AUTHORIZATION_CODE` Zuschusses erforderlich sind.

Felder

- `AuthorizationCode`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 4096 Byte lang, entspricht. Custom string pattern #37

  Ein Autorisierungscode, der in der dritten Phase des Grant-Workflows verwendet werden soll. `AUTHORIZATION_CODE` Dies ist ein einmalig verwendbarer Code, der ungültig wird, sobald er gegen ein Zugriffstoken eingetauscht wird. Daher ist es akzeptabel, diesen Wert als Anforderungsparameter zu verwenden.
- `RedirectUri`— UTF-8-Zeichenfolge, nicht länger als 512 Byte, entspricht. Custom string pattern #41

  Die Umleitungs-URI, zu der der Benutzer bei der Ausgabe eines Autorisierungscodes vom Autorisierungsserver umgeleitet wird. Der URI wird anschließend verwendet, wenn der Autorisierungscode gegen ein Zugriffstoken ausgetauscht wird.

BasicAuthenticationCredentials Struktur

Für die Bereitstellung grundlegender Authentifizierungsdaten, wenn kein `SecretArn` Wert angegeben wird.

Felder

- `Username`— UTF-8-Zeichenfolge, nicht mehr als 512 Byte lang, entspricht. Custom string pattern #37

  Der Benutzername für die Verbindung mit der Datenquelle.
- `Password`— UTF-8-Zeichenfolge, nicht mehr als 512 Byte lang, entspricht. Custom string pattern #33

  Das Passwort für die Verbindung mit der Datenquelle.

OAuth2Struktur der Anmeldeinformationen

Die Anmeldeinformationen, die verwendet werden, wenn der Authentifizierungstyp OAuth2 Authentifizierung ist.

Felder

- `UserManagedClientApplicationClientSecret`— UTF-8-Zeichenfolge, nicht länger als 512 Byte, entspricht. Custom string pattern #38

  Das Client-Geheimnis der Client-Anwendung, wenn die Client-Anwendung vom Benutzer verwaltet wird.
- `AccessToken`— UTF-8-Zeichenfolge, nicht mehr als 4096 Byte lang, entspricht. Custom string pattern #38

  Das Zugriffstoken, das verwendet wird, wenn der Authentifizierungstyp ist. OAuth2
- `RefreshToken`— UTF-8-Zeichenfolge, nicht mehr als 4096 Byte lang, entspricht. Custom string pattern #38

  Das Aktualisierungstoken, das verwendet wird, wenn der Authentifizierungstyp ist. OAuth2
- `JwtToken`— UTF-8-Zeichenfolge, nicht länger als 8000 Byte, entspricht. Custom string pattern #39

  Das JSON Web Token (JWT), das verwendet wird, wenn der Authentifizierungstyp ist. OAuth2

Operationen

- CreateConnection Aktion (Python: create_connection)
- DeleteConnection Aktion (Python: delete_connection)
- GetConnection Aktion (Python: get_connection)
- GetConnections Aktion (Python: get_connections)
- UpdateConnection Aktion (Python: update_connection)
- TestConnection Aktion (Python: test_connection)
- BatchDeleteConnection Aktion (Python: batch_delete_connection)

CreateConnection Aktion (Python: create_connection)

Erstellt eine Verbindungsdefinition im Data Catalog.

Verbindungen, die zum Erstellen von Verbundressourcen verwendet werden, erfordern die IAM-Berechtigung `glue:PassConnection`.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Verbindung erstellt wird. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `ConnectionInput` – Erforderlich: Ein ConnectionInput-Objekt.

  Ein `ConnectionInput`-Objekt, das die zu erstellende Verbindung definiert.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags, die Sie der Verbindung zuweisen.

Antwort

- `CreateConnectionStatus` – UTF-8-Zeichenfolge (zulässige Werte: `READY` | `IN_PROGRESS` | `FAILED`).

  Der Status der Anfrage zur Verbindungserstellung. Die Anfrage kann bei bestimmten Authentifizierungstypen einige Zeit in Anspruch nehmen, z. B. beim Herstellen einer OAuth Verbindung mit Tokenaustausch über VPC.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`

DeleteConnection Aktion (Python: delete_connection)

Löscht eine Verbindung aus dem Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Verbindung befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu löschenden Verbindung.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`

GetConnection Aktion (Python: get_connection)

Ruft eine Verbindungsdefinition aus dem Data Catalog ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Verbindung befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindungsdefinition, die abgerufen werden soll.
- `HidePassword` – Boolesch.

  Damit können Sie die Verbindungsmetadaten abrufen, ohne das Passwort zurückzugeben. Beispielsweise verwendet die AWS Glue Konsole dieses Flag, um die Verbindung abzurufen, und zeigt das Passwort nicht an. Legen Sie diesen Parameter fest, wenn der Anrufer möglicherweise nicht berechtigt ist, den AWS KMS Schlüssel zum Entschlüsseln des Kennworts zu verwenden, er jedoch berechtigt ist, auf die übrigen Verbindungseigenschaften zuzugreifen.
- `ApplyOverrideForComputeEnvironment` – UTF-8-Zeichenfolge (zulässige Werte: `SPARK` | `ATHENA` | `PYTHON`).

  Gibt für Verbindungen, die in mehreren Diensten verwendet werden können, die Rückgabe von Eigenschaften für die angegebene Rechenumgebung an.

Antwort

- `Connection` – Ein Verbindung-Objekt.

  Die angeforderte Verbindungsdefinition.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

GetConnections Aktion (Python: get_connections)

Ruft eine Liste von Verbindungsdefinitionen aus dem Data Catalog ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Verbindungen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `Filter` – Ein GetConnectionsFilter-Objekt.

  Ein Filter, der steuert, welche Verbindungen zurückgegeben werden.
- `HidePassword` – Boolesch.

  Damit können Sie die Verbindungsmetadaten abrufen, ohne das Passwort zurückzugeben. Beispielsweise verwendet die AWS Glue Konsole dieses Flag, um die Verbindung abzurufen, und zeigt das Passwort nicht an. Legen Sie diesen Parameter fest, wenn der Anrufer möglicherweise nicht berechtigt ist, den AWS KMS Schlüssel zum Entschlüsseln des Kennworts zu verwenden, er jedoch berechtigt ist, auf die übrigen Verbindungseigenschaften zuzugreifen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl von Verbindungen, die in einer Antwort zurückgegeben werden sollen.

Antwort

- `ConnectionList` – Ein Array mit Verbindung-Objekten.

Eine Liste von angeforderten Verbindungsdefinitionen.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die Liste der zurückgegebenen Verbindungen nicht die letzten gefilterten Verbindungen enthält.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

UpdateConnection Aktion (Python: update_connection)

Aktualisiert eine Verbindungsdefinition im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Verbindung befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindungsdefinition, die aktualisiert werden soll.

- `ConnectionInput` – Erforderlich: Ein ConnectionInput-Objekt.

  Ein `ConnectionInput`-Objekt, das die jeweilige Verbindung neu definiert.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

TestConnection Aktion (Python: test_connection)

Testet eine Verbindung zu einem Dienst, um die von Ihnen angegebenen Anmeldeinformationen für den Dienst zu überprüfen.

Sie können entweder einen vorhandenen Verbindungsnamen oder eine Eingabe `TestConnectionInput` zum Testen einer nicht vorhandenen Verbindung angeben. Wenn Sie beide gleichzeitig angeben, wird ein Fehler verursacht.

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Anforderung

- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Optional. Der Name der zu testenden Verbindung. Wenn nur der Name angegeben wird, stellt der Vorgang die Verbindung her und verwendet sie zum Testen.
- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID, unter der sich die Verbindung befindet.
- `TestConnectionInput` – Ein TestConnectionInput-Objekt.

  Eine Struktur, die verwendet wird, um das Testen einer Verbindung zu einem Dienst anzugeben.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`
- `FederationSourceException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConflictException`
- `InternalServiceException`

BatchDeleteConnection Aktion (Python: batch_delete_connection)

Löscht eine Liste von Verbindungsdefinitionen aus dem Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Verbindungen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `ConnectionNameList` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 25 Zeichenfolgen.

  Eine Liste der Namen der Verbindungen, die gelöscht werden sollen.

Antwort

- `Succeeded` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Namen der Verbindungsdefinitionen, die erfolgreich gelöscht wurden.
- `Errors` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

Jeder Wert ist ein An ErrorDetail-Objekt.

Eine Zuweisung der Namen von Verbindungen, die nicht erfolgreich gelöscht wurden, zu Fehlerdetails.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`

API für Verbindungstypen

Die Verbindungstyp-API beschreibt AWS Glue APIs im Zusammenhang mit der Beschreibung von Verbindungstypen.

Verwaltung der Verbindungen APIs

- DescribeConnectionType Aktion (Python: describe_connection_type)
- ListConnectionTypes Aktion (Python: list_connection_types)
- ConnectionTypeBrief Struktur
- ConnectionTypeVariant Struktur

DescribeConnectionType Aktion (Python: describe_connection_type)

Die `DescribeConnectionType` API bietet vollständige Informationen zu den unterstützten Optionen für einen bestimmten Verbindungstyp in. AWS Glue

Anforderung

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Verbindungstyps, der beschrieben werden soll.

Antwort

- `ConnectionType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Verbindungstyps.

- `Description`— UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine Beschreibung des Verbindungstyps.

- `Capabilities` – Ein Funktionen-Objekt.

  Die unterstützten Authentifizierungstypen, Datenschnittstellentypen (Rechenumgebungen) und Datenoperationen des Connectors.

- `ConnectionProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Verbindungseigenschaften, die in allen Computerumgebungen üblich sind.

- `ConnectionOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Gibt Eigenschaften zurück, die beim Erstellen einer Verbindung in der festgelegt werden können`ConnectionInput.ConnectionProperties`. `ConnectionOptions`definiert Parameter, die in einem Spark-ETL-Skript in der Verbindungsoptionsübersicht festgelegt werden können, die an einen Datenrahmen übergeben wird.

- `AuthenticationConfiguration` – Ein AuthConfiguration-Objekt.

  Der für die Verbindung verwendete Authentifizierungstyp.

- `ComputeEnvironmentConfigurations` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

Jeder Wert ist ein A ComputeEnvironmentConfiguration-Objekt.

Die Computerumgebungen, die von der Verbindung unterstützt werden.

- `PhysicalConnectionRequirements` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Physikalische Anforderungen für eine Verbindung, z. B. VPC-, Subnetz- und Sicherheitsgruppenspezifikationen.

- `AthenaConnectionProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Spezifische Verbindungseigenschaften für die Athena-Rechenumgebung.

- `PythonConnectionProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Verbindungseigenschaften, die für die Python-Rechenumgebung spezifisch sind.

- `SparkConnectionProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Spezifische Verbindungseigenschaften für die Spark-Rechenumgebung.

Fehler

- `ValidationException`
- `InvalidInputException`
- `InternalServiceException`

ListConnectionTypes Aktion (Python: list_connection_types)

Die `ListConnectionTypes` API bietet einen Erkennungsmechanismus zum Ermitteln verfügbarer Verbindungstypen in. AWS Glue Die Antwort enthält eine Liste von Verbindungstypen mit detaillierten Informationen darüber, was für jeden Verbindungstyp unterstützt wird. Bei den aufgelisteten Verbindungstypen handelt es sich um die unterstützten Optionen für den `ConnectionType` Wert in der `CreateConnection` API.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.
- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `ConnectionTypes` – Ein Array mit ConnectionTypeBrief-Objekten.

  Eine Liste von `ConnectionTypeBrief` Objekten mit kurzen Informationen zu den unterstützten Verbindungstypen.
- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungstoken, wenn das aktuelle Listensegment nicht das letzte ist.

Fehler

- `InternalServiceException`

ConnectionTypeBrief Struktur

Kurze Informationen zu einem unterstützten Verbindungstyp, der von der `ListConnectionTypes` API zurückgegeben wurde.

Felder

- `ConnectionType`— UTF-8-Zeichenfolge (gültige Werte: `JDBC` | | | `SFTP` | `MONGODB` | `KAFKA` | | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | | `GOOGLESHEETS` | `GOOGLEANALYTICS4` | `SERVICENOW` | `MARKETO` | | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `SLACK` | `STRIPE` `INTERCOM` |`SNAPCHATADS`).

  Der Name des Verbindungstyps.

- `DisplayName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der menschenlesbare Name für den Verbindungstyp, der in der AWS Glue Konsole angezeigt wird.

- `Vendor` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Name des Anbieters oder Anbieters, der diesen Verbindungstyp erstellt oder verwaltet.

- `Description`— UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine Beschreibung des Verbindungstyps.

- `Categories` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Kategorien, zu denen dieser Verbindungstyp gehört. Mithilfe von Kategorien können Benutzer anhand ihrer Anwendungsfälle geeignete Verbindungstypen filtern und finden.

- `Capabilities` – Ein Funktionen-Objekt.

  Die unterstützten Authentifizierungstypen, Datenschnittstellentypen (Rechenumgebungen) und Datenoperationen des Connectors.

- `LogoUrl` – UTF-8-Zeichenfolge.

  Die URL des Logos, das einem Verbindungstyp zugeordnet ist.

- `ConnectionTypeVariants` – Ein Array mit ConnectionTypeVariant-Objekten.

Eine Liste der für diesen Verbindungstyp verfügbaren Varianten. Verschiedene Varianten bieten möglicherweise spezielle Konfigurationen für bestimmte Anwendungsfälle oder Implementierungen desselben allgemeinen Verbindungstyps.

ConnectionTypeVariant Struktur

Stellt eine Variante eines Verbindungstyps im AWS Glue Datenkatalog dar. Verbindungstypvarianten bieten spezifische Konfigurationen und Verhaltensweisen für verschiedene Implementierungen desselben allgemeinen Verbindungstyps.

Felder

- `ConnectionTypeVariantName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Die eindeutige Kennung für die Verbindungstypvariante. Dieser Name wird intern verwendet, um die spezifische Variante eines Verbindungstyps zu identifizieren.
- `DisplayName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der für Menschen lesbare Name für die Variante des Verbindungstyps, der in der AWS Glue Konsole angezeigt wird.
- `Description`— UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine ausführliche Beschreibung der Variante des Verbindungstyps, einschließlich ihres Zwecks, ihrer Anwendungsfälle und aller spezifischen Konfigurationsanforderungen.
- `LogoUrl` – UTF-8-Zeichenfolge.

  Die URL des Logos, das einer Verbindungstypvariante zugeordnet ist.

Datentypen

- Struktur der Validierung
- AuthConfiguration Struktur
- Struktur der Fähigkeiten
- Struktur der Immobilie
- AllowedValue Struktur

- ComputeEnvironmentConfiguration Struktur

Struktur der Validierung

Definiert, wie eine Überprüfung einer Verbindungseigenschaft durchgeführt wird.

Felder

- `ValidationType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `REGEX` | `RANGE`).

  Die Art der durchzuführenden Überprüfung, z. `REGEX` B.
- `Patterns` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Mustern, die für die Validierung gelten.
- `Description`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Eine Beschreibung für die Validierung.
- `MaxLength` – Zahl (Ganzzahl).

  Eine maximale Länge für eine Verbindungseigenschaft einer Zeichenfolge.
- `Maximum` – Zahl (Ganzzahl).

  Ein Maximalwert bei der Angabe eines `RANGE` Überprüfungstyps.
- `Minimum` – Zahl (Ganzzahl).

  Ein Mindestwert bei der Angabe eines `RANGE` Überprüfungstyps.

AuthConfiguration Struktur

Die Authentifizierungskonfiguration für eine Verbindung, die von der `DescribeConnectionType` API zurückgegeben wurde.

Felder

- `AuthenticationType` – Erforderlich: Ein Eigenschaft-Objekt.

  Die Art der Authentifizierung für eine Verbindung.
- `SecretArn` – Ein Eigenschaft-Objekt.

Der Amazon-Ressourcenname (ARN) für den Secrets Manager.

- `OAuth2Properties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Eine Zuordnung von Schlüssel-Wert-Paaren für die OAuth2 Eigenschaften. Jeder Wert ist ein `Property` Objekt.

- `BasicAuthenticationProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Eine Zuordnung von Schlüssel-Wert-Paaren für die OAuth2 Eigenschaften. Jeder Wert ist ein `Property` Objekt.

- `CustomAuthenticationProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Eine Zuordnung von Schlüssel-Wert-Paaren für die benutzerdefinierten Authentifizierungseigenschaften. Jeder Wert ist ein `Property` Objekt.

Struktur der Fähigkeiten

Gibt die unterstützten Authentifizierungstypen an, die von der `DescribeConnectionType` API zurückgegeben werden.

Felder

- `SupportedAuthenticationTypes` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der unterstützten Authentifizierungstypen.

- `SupportedDataOperations` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der unterstützten Datenoperationen.
- `SupportedComputeEnvironments` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der unterstützten Rechenumgebungen.

Struktur der Immobilie

Ein Objekt, das einen Verbindungstyp für eine Rechenumgebung definiert.

Felder

- `Name`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Name der Eigenschaft.
- `Description`— Erforderlich: UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine Beschreibung der Immobilie.
- `Required` – Erforderlich: Boolean.

  Gibt an, ob die Eigenschaft erforderlich ist.
- `DefaultValue` – UTF-8-Zeichenfolge.

  Der Standardwert für die Eigenschaft.
- `PropertyTypes` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Beschreibt den Eigenschaftstyp.
- `AllowedValues` – Ein Array mit AllowedValue-Objekten.

  Eine Liste von `AllowedValue` Objekten, die die für die Eigenschaft zulässigen Werte darstellen.
- `DataOperationScopes` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt an, welche Datenoperationen auf die Eigenschaft anwendbar sind.

AllowedValue Struktur

Ein Objekt, das einen für eine Eigenschaft zulässigen Wert darstellt.

Felder

- `Description`— UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine Beschreibung des zulässigen Werts.
- `Value`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der für die Eigenschaft zulässige Wert.

ComputeEnvironmentConfiguration Struktur

Ein Objekt, das die Konfiguration für eine Rechenumgebung (wie Spark, Python oder Athena) enthält, die von der `DescribeConnectionType` API zurückgegeben wird.

Felder

- `Name`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein Name für die Konfiguration der Rechenumgebung.
- `Description`— Erforderlich: UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Eine Beschreibung der Rechenumgebung.
- `ComputeEnvironment` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `SPARK` | `ATHENA` | `PYTHON`).

  Der Typ der Datenverarbeitungsumgebung.
- `SupportedAuthenticationTypes` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die unterstützten Authentifizierungstypen für die Rechenumgebung.
- `ConnectionOptions` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist ein A Eigenschaft-Objekt.

  Die Parameter, die als Verbindungsoptionen für die Rechenumgebung verwendet werden.
- `ConnectionPropertyNameOverrides` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Byte lang ist.

Der Name der Verbindungseigenschaft überschreibt für die Rechenumgebung.

- `ConnectionOptionNameOverrides` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Byte lang ist.

  Der Name der Verbindungsoption überschreibt für die Rechenumgebung.

- `ConnectionPropertiesRequiredOverrides` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die Verbindungseigenschaften, die als Überschreibungen für die Rechenumgebung erforderlich sind.

- `PhysicalConnectionPropertiesRequired` – Boolesch.

  Gibt an, ob `PhysicalConnectionProperties` sie für die Rechenumgebung erforderlich sind.

Verbindungsmetadaten und Vorschau-API

In der folgenden Verbindung APIs werden Operationen zur Beschreibung von Verbindungsmetadaten beschrieben.

Datentypen

- Struktur der Entität
- Struktur des Feldes

Struktur der Entität

Eine Entität, die von einer bestimmten Person unterstützt wird`ConnectionType`.

Felder

- `EntityName` – UTF-8-Zeichenfolge.

  Der Name der Entität.

- `Label` – UTF-8-Zeichenfolge.

  Für die Entität verwendetes Label.

- `IsParentEntity` – Boolesch.

  Ein boolescher Wert, mit dessen Hilfe bestimmt werden kann, ob Unterobjekte aufgelistet werden können.

- `Description` – UTF-8-Zeichenfolge.

  Eine Beschreibung der Entität.

- `Category` – UTF-8-Zeichenfolge.

  Der Typ der Entitäten, die in der Antwort enthalten sind. Dieser Wert hängt von der Quellverbindung ab. Dies ist beispielsweise `SObjects` für Salesforce und/oder `databases` `schemas` oder `tables` für Quellen wie Amazon Redshift.

- `CustomProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine optionale Zuordnung von Schlüsseln, die für eine Entität von einem Konnektor zurückgegeben werden können.

Struktur des Feldes

Das `Field` Objekt enthält Informationen über die verschiedenen Eigenschaften, die einem Feld im Konnektor zugeordnet sind.

Felder

- `FieldName` – UTF-8-Zeichenfolge.

  Ein eindeutiger Bezeichner für das Feld.

- `Label` – UTF-8-Zeichenfolge.

  Eine lesbare Bezeichnung, die für das Feld verwendet wird.

- `Description` – UTF-8-Zeichenfolge.

  Eine Beschreibung des Felds.

- `FieldType`— UTF-8-Zeichenfolge (gültige Werte: `INT` | | | `SMALLINT` | `BIGINT` | `FLOAT` | `LONG` | `DATE` | | `BOOLEAN` | `MAP` | `ARRAY` | `STRING` | `TIMESTAMP` | `DECIMAL` | | `BYTE` `SHORT` `DOUBLE` |`STRUCT`).

  Der Datentyp im Feld.

- `IsPrimaryKey` – Boolesch.

  Gibt an, ob dieses Feld als Primärschlüssel für die angegebene Entität verwendet werden kann.

- `IsNullable` – Boolesch.

  Gibt an, ob für dieses Feld NULL-Werte zulässig sind oder nicht.

- `IsRetrievable` – Boolesch.

  Gibt an, ob dieses Feld der Select-Klausel der SQL-Abfrage hinzugefügt werden kann oder ob es abrufbar ist oder nicht.

- `IsFilterable` – Boolesch.

  Gibt an, ob dieses Feld in einer Filterklausel (`WHERE`Klausel) einer SQL-Anweisung beim Abfragen von Daten verwendet werden kann.

- `IsPartitionable` – Boolesch.

  Gibt an, ob ein bestimmtes Feld zur Partitionierung der an SaaS gestellten Abfrage verwendet werden kann.

- `IsCreateable` – Boolesch.

  Gibt an, ob dieses Feld als Teil eines Ziel-Schreibvorgangs erstellt werden kann.

- `IsUpdateable` – Boolesch.

  Gibt an, ob dieses Feld im Rahmen eines Ziel-Schreibvorgangs aktualisiert werden kann.

- `IsUpsertable` – Boolesch.

  Gibt an, ob dieses Feld im Rahmen eines Ziel-Schreibvorgangs geändert werden kann.

- `IsDefaultOnCreate` – Boolesch.

  Gibt an, ob dieses Feld automatisch aufgefüllt wird, wenn das Objekt erstellt wird, z. B. ein Zeitstempel vom Typ Created at.

- `SupportedValues` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der unterstützten Werte für das Feld.

- `SupportedFilterOperators` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt an, welche Filteroperatoren für dieses Feld unterstützt werden.

- `ParentField` – UTF-8-Zeichenfolge.

  Ein übergeordneter Feldname für ein verschachteltes Feld.

- `NativeDataType` – UTF-8-Zeichenfolge.

  Der von der SaaS-API zurückgegebene Datentyp, z. B. „Picklist“ oder „Textarea“ von Salesforce.

- `CustomProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Optionale Zuordnung von Schlüsseln, die zurückgegeben werden können.

Operationen

- ListEntities Aktion (Python: list_entities)
- DescribeEntity Aktion (Python: describe_entity)
- GetEntityRecords Aktion (Python: get_entity_records)

ListEntities Aktion (Python: list_entities)

Gibt die verfügbaren Entitäten zurück, die vom Verbindungstyp unterstützt werden.

Anforderung

- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Ein Name für die Verbindung, für die Anmeldeinformationen erforderlich sind, um einen beliebigen Verbindungstyp abzufragen.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID des Katalogs, der die Verbindung enthält. Dies kann Null sein. Standardmäßig ist die AWS Konto-ID die Katalog-ID.

- `ParentEntityName` – UTF-8-Zeichenfolge.

  Name der übergeordneten Entität, für die Sie die untergeordneten Entität auflisten möchten. Dieser Parameter verwendet einen vollständig qualifizierten Pfad der Entität, um die untergeordneten Entitäten aufzulisten.

- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.

- `DataStoreApiVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #23.

  Die API-Version des SaaS-Connectors.

Antwort

- `Entities` – Ein Array mit Entität-Objekten.

  Eine Liste von `Entity`-Objekten.

- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungstoken, das vorhanden ist, wenn das aktuelle Segment nicht das letzte ist.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

- `GlueEncryptionException`
- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

DescribeEntity Aktion (Python: describe_entity)

Enthält Details zu der Entität, die mit dem Verbindungstyp verwendet wird, sowie eine Beschreibung des Datenmodells für jedes Feld in der ausgewählten Entität.

Die Antwort umfasst alle Felder, aus denen die Entität besteht.

Anforderung

- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung, die die Anmeldeinformationen für den Verbindungstyp enthält.
- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID des Katalogs, der die Verbindung enthält. Dies kann Null sein. Standardmäßig ist die AWS Konto-ID die Katalog-ID.
- `EntityName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name der Entität, die Sie anhand des Verbindungstyps beschreiben möchten.
- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.
- `DataStoreApiVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #23.

  Die Version der API, die für den Datenspeicher verwendet wird.

Antwort

- `Fields` – Ein Array mit Feld-Objekten.

Beschreibt die Felder für diese Connector-Entität. Dies ist die Liste der `Field` Objekte. `Field`dist einer Spalte in einer Datenbank sehr ähnlich. Das `Field` Objekt enthält Informationen über verschiedene Eigenschaften, die Feldern im Konnektor zugeordnet sind.

- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungstoken, das vorhanden ist, wenn das aktuelle Segment nicht das letzte ist.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`
- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

GetEntityRecords Aktion (Python: get_entity_records)

Diese API wird verwendet, um Vorschaudaten von einem bestimmten Verbindungstyp oder aus einem nativen Amazon S3 S3-basierten AWS Glue Datenkatalog abzufragen.

Gibt Datensätze als Array von JSON-Blobs zurück. Jeder Datensatz wird mithilfe von Jackson auf der JsonNode Grundlage des von der API definierten Feldtyps formatiert. `DescribeEntity`

Spark-Konnektoren generieren Schemas gemäß derselben Datentypzuordnung wie in der `DescribeEntity` API. Spark-Konnektoren konvertieren Daten in die entsprechenden Datentypen, die dem Schema entsprechen, wenn Zeilen zurückgegeben werden.

Anforderung

- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Verbindung, die die Anmeldeinformationen für den Verbindungstyp enthält.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID des Katalogs, der die Verbindung enthält. Dies kann Null sein. Standardmäßig ist die AWS Konto-ID die Katalog-ID.

- `EntityName` – Erforderlich: UTF-8-Zeichenfolge.

  Name der Entität, für die wir die Vorschaudaten des angegebenen Verbindungstyps abfragen möchten.

- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.

- `DataStoreApiVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #23.

  Die API-Version des SaaS-Connectors.

- `ConnectionOptions` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 256 Byte lang ist und der entspricht. Custom string pattern #18

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 256 Byte lang ist und der entspricht. Custom string pattern #17

  Konnektoroptionen, die zum Abfragen der Daten erforderlich sind.

- `FilterPredicate`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 100000 Byte lang.

  Ein Filterprädikat, das Sie in der Abfrageanforderung anwenden können.

- `Limit`— Erforderlich: Zahl (lang), nicht weniger als 1 oder mehr als 1000.

  Beschränkt die Anzahl der mit der Anfrage abgerufenen Datensätze.

- `OrderBy` – UTF-8-Zeichenfolge.

  Ein Parameter, der die Antwortvorschaudaten sortiert.

- `SelectedFields`— Ein Array von UTF-8-Zeichenfolgen, nicht weniger als 1 oder mehr als 1000 Zeichenketten.

Liste der Felder, die wir als Teil der Vorschaudaten abrufen möchten.

Antwort

- `Records` – Eine Reihe von Strukturen.

  Eine Liste der angeforderten -Objekte.
- `NextToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum Custom string pattern #11.

  Ein Fortsetzungstoken, das vorhanden ist, wenn das aktuelle Segment nicht das letzte ist.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`
- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

Benutzerdefinierte Funktion (API)

Die API für benutzerdefinierte Funktionen beschreibt AWS Glue Datentypen und Operationen, die bei der Arbeit mit Funktionen verwendet werden.

Datentypen

- UserDefinedFunction Struktur
- UserDefinedFunctionInput Struktur

UserDefinedFunction Struktur

Stellt das Äquivalent einer benutzerdefinierten Hive-Funktion (`UDF`) dar.

Felder

- `FunctionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Funktion.

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, die die Funktion enthält

- `ClassName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Java-Klasse mit dem Funktionscode.

- `OwnerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Eigentümer der Funktion.

- `OwnerType` – UTF-8-Zeichenfolge (zulässige Werte: `USER` | `ROLE` | `GROUP`).

  Der Typ des Eigentümers.

- `CreateTime` – Zeitstempel.

  Die Uhrzeit, zu der die Funktion erstellt wurde.

- `ResourceUris` – Ein Array mit ResourceUri-Objekten, nicht mehr als 1000 Strukturen.

  Die Ressource URIs für die Funktion.

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Funktion befindet.

UserDefinedFunctionInput Struktur

Eine Struktur, mit der eine benutzerdefinierte Funktion angelegt oder aktualisiert wird.

Felder

- `FunctionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Funktion.

- `ClassName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Java-Klasse mit dem Funktionscode.

- `OwnerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Eigentümer der Funktion.

- `OwnerType` – UTF-8-Zeichenfolge (zulässige Werte: `USER` | `ROLE` | `GROUP`).

  Der Typ des Eigentümers.

- `ResourceUris` – Ein Array mit ResourceUri-Objekten, nicht mehr als 1000 Strukturen.

  Die Ressource URIs für die Funktion.

Operationen

- CreateUserDefinedFunction Aktion (Python: create_user_defined_function)
- UpdateUserDefinedFunction Aktion (Python: update_user_defined_function)
- DeleteUserDefinedFunction Aktion (Python: delete_user_defined_function)
- GetUserDefinedFunction Aktion (Python: get_user_defined_function)
- GetUserDefinedFunctions Aktion (Python: get_user_defined_functions)

CreateUserDefinedFunction Aktion (Python: create_user_defined_function)

Erstellt eine neue Funktionsdefinition im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die ID des Data Catalog, in dem die Funktion erstellt wird. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Name der Katalogdatenbank, in der die Funktion angelegt wird.

- `FunctionInput` – Erforderlich: Ein UserDefinedFunctionInput-Objekt.

  Das `FunctionInput`-Objekt, das die im Data Catalog zu erstellende Funktion definiert.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`

UpdateUserDefinedFunction Aktion (Python: update_user_defined_function)

Aktualisiert eine bestehende Funktionsdefinition im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die zu aktualisierende Funktion befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die zu aktualisierende Funktion befindet.
- `FunctionName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Funktion.
- `FunctionInput` – Erforderlich: Ein UserDefinedFunctionInput-Objekt.

  Ein `FunctionInput`-Objekt, das die Funktion im Data Catalog neu definiert.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteUserDefinedFunction Aktion (Python: delete_user_defined_function)

Löscht eine bestehende Funktionsdefinition im Data Catalog.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die zu löschende Funktion befindet. Wenn keine angegeben wird, wird standardmäßig die Konto-ID verwendet. AWS
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Katalogdatenbank, in der sich die Funktion befindet.

- `FunctionName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Funktionsdefinition, die gelöscht werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetUserDefinedFunction Aktion (Python: get_user_defined_function)

Ruft eine angegebene Funktionsdefinition aus dem Data Catalog ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die abzurufende Funktion befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Funktion befindet.

- `FunctionName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Funktion.

Antwort

- `UserDefinedFunction` – Ein UserDefinedFunction-Objekt.

  Die angeforderte Funktionsdefinition.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetUserDefinedFunctions Aktion (Python: get_user_defined_functions)

Ruft mehrere Funktionsdefinition aus dem Data Catalog ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die abzurufenden Funktionen befinden. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.
- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Katalogdatenbank, in der sich die Funktionen befinden. Wenn keine bereitgestellt wird, werden Funktionen aus allen Datenbanken im Katalog zurückgegeben.
- `Pattern` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein optionaler Musterstring für Funktionsnamen, der die zurückgegebenen Funktionsdefinitionen filtert.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Die maximale Anzahl der Funktionen, die in einer Antwort zurückgegeben werden können.

Antwort

- `UserDefinedFunctions` – Ein Array mit UserDefinedFunction-Objekten.

  Eine Liste der angeforderten Funktionsdefinitionen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die Liste der zurückgegebenen Funktionen nicht die zuletzt angeforderte Funktion enthält.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

Einen Athena Katalog importieren nach AWS Glue

Die Migrations-API beschreibt AWS Glue Datentypen und Vorgänge, die mit der Migration eines Athena Datenkatalogs zu zu tun haben. AWS Glue

Datentypen

- CatalogImportStatus Struktur

CatalogImportStatus Struktur

Eine Struktur, die Migrationsstatusinformationen enthält.

Felder

- `ImportCompleted` – Boolesch.

`True`, wenn die Migration abgeschlossen ist, andernfalls `False`.

- `ImportTime` – Zeitstempel.

  Der Zeitpunkt des Beginns der Migration.

- `ImportedBy` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Person, die die Migration eingeleitet hat.

Operationen

- ImportCatalogToGlue Aktion (Python: import_catalog_to_glue)
- GetCatalogImportStatus Aktion (Python: get_catalog_import_status)

ImportCatalogToGlue Aktion (Python: import_catalog_to_glue)

Importiert einen vorhandenen Amazon Athena Athena-Datenkatalog in AWS Glue.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs, der importiert werden soll. Derzeit sollte dies die AWS Konto-ID sein.

Antwort

- Keine Antwortparameter.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`

GetCatalogImportStatus Aktion (Python: get_catalog_import_status)

Ruft den Status einer Migrationsoperation ab.

Anforderung

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Katalogs, der migriert werden soll. Derzeit sollte dies die Konto-ID sein. AWS

Antwort

- `ImportStatus` – Ein CatalogImportStatus-Objekt.

  Der Status der angegebenen Katalogmigration.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`

Tabellenoptimierer-API

Die Tabellenoptimierer-API beschreibt die AWS Glue API zur Aktivierung der Komprimierung zur Verbesserung der Leseleistung.

Datentypen

- TableOptimizer Struktur
- TableOptimizerConfiguration Struktur
- TableOptimizerVpcConfiguration Struktur
- CompactionConfiguration Struktur
- IcebergCompactionConfiguration Struktur
- TableOptimizerRun Struktur
- BatchGetTableOptimizerEntry Struktur
- BatchTableOptimizer Struktur
- BatchGetTableOptimizerError Struktur
- RetentionConfiguration Struktur

- IcebergRetentionConfiguration Struktur
- OrphanFileDeletionConfiguration Struktur
- IcebergOrphanFileDeletionConfiguration Struktur
- CompactionMetrics Struktur
- RetentionMetrics Struktur
- OrphanFileDeletionMetrics Struktur
- IcebergCompactionMetrics Struktur
- IcebergRetentionMetrics Struktur
- IcebergOrphanFileDeletionMetrics Struktur
- RunMetrics Struktur

TableOptimizer Struktur

Enthält Details zu einem Optimierer für eine Tabelle.

Felder

- `type` – UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers. Die gültigen Werte sind:

  - `compaction`: zur Verwaltung der Komprimierung mit einem Tabellenoptimierer.
  - `retention`: zur Verwaltung der Aufbewahrung von Snapshots mit einem Tabellenoptimierer.
  - `orphan_file_deletion`: zur Verwaltung des Löschens verwaister Dateien mit einem Tabellenoptimierer.
- `configuration` – Ein TableOptimizerConfiguration-Objekt.

  Ein `TableOptimizerConfiguration`-Objekt, das beim Erstellen oder Aktualisieren eines Tabellenoptimierers angegeben wurde.
- `lastRun` – Ein TableOptimizerRun-Objekt.

  Ein `TableOptimizerRun`-Objekt, das den letzten Lauf des Tabellenoptimierers darstellt.

TableOptimizerConfiguration Struktur

Enthält Details zur Konfiguration eines Tabellenoptimierers. Sie übergeben diese Konfiguration, wenn Sie einen Tabellenoptimierer erstellen oder aktualisieren.

Felder

- `roleArn` – UTF-8-Zeichenfolge, nicht weniger als 20 oder mehr als 2 048 Bytes lang, passend zum Single-line string pattern.

  Eine vom Aufrufer übergebene Rolle, die dem Service die Erlaubnis erteilt, die mit dem Optimierer verknüpften Ressourcen im Namen des Aufrufers zu aktualisieren.

- `enabled` – Boolesch.

  Ob die Tabellenoptimierung aktiviert ist.

- `vpcConfiguration` – Ein TableOptimizerVpcConfiguration-Objekt.

  Ein `TableOptimizerVpcConfiguration` Objekt, das die VPC-Konfiguration für einen Tabellenoptimierer darstellt.

  Diese Konfiguration ist erforderlich, um die Optimierung von Tabellen durchzuführen, die sich in einer Kunden-VPC befinden.

- `compactionConfiguration` – Ein CompactionConfiguration-Objekt.

  Die Konfiguration für einen Verdichtungsoptimierer. Diese Konfiguration definiert, wie Datendateien in Ihrer Tabelle komprimiert werden, um die Abfrageleistung zu verbessern und die Speicherkosten zu senken.

- `retentionConfiguration` – Ein RetentionConfiguration-Objekt.

  Die Konfiguration für einen Snapshot-Aufbewahrungsoptimierer.

- `orphanFileDeletionConfiguration` – Ein OrphanFileDeletionConfiguration-Objekt.

  Die Konfiguration für einen Optimizer für das Löschen verwaister Dateien.

TableOptimizerVpcConfiguration Struktur

Ein Objekt, das die VPC-Konfiguration für einen Tabellenoptimierer beschreibt.

Diese Konfiguration ist erforderlich, um die Optimierung von Tabellen durchzuführen, die sich in einer Kunden-VPC befinden.

Felder

- `glueConnectionName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der AWS Glue Verbindung, die für die VPC für den Tabellenoptimierer verwendet wird.

CompactionConfiguration Struktur

Die Konfiguration für einen Verdichtungsoptimierer. Diese Konfiguration definiert, wie Datendateien in Ihrer Tabelle komprimiert werden, um die Abfrageleistung zu verbessern und die Speicherkosten zu senken.

Felder

- `icebergConfiguration` – Ein IcebergCompactionConfiguration-Objekt.

  Die Konfiguration für einen Iceberg-Verdichtungsoptimierer.

IcebergCompactionConfiguration Struktur

Die Konfiguration für einen Iceberg-Verdichtungsoptimierer. Diese Konfiguration definiert Parameter für die Optimierung des Layouts von Datendateien in Iceberg-Tabellen.

Felder

- `strategy` – UTF-8-Zeichenfolge (zulässige Werte: `binpack="BINPACK"` | `sort="SORT"` | `z-order="ZORDER"`).

  Die Strategie, die für die Verdichtung verwendet werden soll. Gültige Werte für sind:

  - `binpack`: Kombiniert kleine Dateien zu größeren Dateien und zielt in der Regel auf Größen über 100 MB ab, wobei alle ausstehenden Löschungen angewendet werden. Dies ist die empfohlene Komprimierungsstrategie für die meisten Anwendungsfälle.
  - `sort`: Organisiert Daten auf der Grundlage bestimmter Spalten, die während der Komprimierung hierarchisch sortiert werden, wodurch die Abfrageleistung für gefilterte Operationen verbessert wird. Diese Strategie wird empfohlen, wenn Ihre Abfragen häufig nach bestimmten Spalten

filtern. Um diese Strategie zu verwenden, müssen Sie zunächst mithilfe der Tabelleneigenschaft eine Sortierreihenfolge in Ihren `sort_order` Iceberg-Tabelleneigenschaften definieren.

- `z-order`: Optimiert die Datenorganisation, indem mehrere Attribute zu einem einzigen Skalarwert zusammengefasst werden, der für die Sortierung verwendet werden kann, wodurch effiziente Abfragen über mehrere Dimensionen hinweg ermöglicht werden. Diese Strategie wird empfohlen, wenn Sie Daten über mehrere Dimensionen gleichzeitig abfragen müssen. Um diese Strategie zu verwenden, müssen Sie zunächst mithilfe der Tabelleneigenschaft eine Sortierreihenfolge in Ihren `sort_order` Iceberg-Tabelleneigenschaften definieren.

Wenn keine Eingabe bereitgestellt wird, wird der Standardwert 'binpack' verwendet.

TableOptimizerRun Struktur

Enthält Details zur Ausführung eines Tabellenoptimierers.

Felder

- `eventType` – UTF-8-Zeichenfolge (zulässige Werte: `starting="STARTING"` | `completed="COMPLETED"` | `failed="FAILED"` | `in_progress="IN_PROGRESS"`).

  Ein Ereignistyp, der den Status der Ausführung des Tabellenoptimierers darstellt.

- `startTimestamp` – Zeitstempel.

  Stellt den Epoch-Zeitstempel für den Beginn des Verdichtungsauftrags in Lake Formation dar.

- `endTimestamp` – Zeitstempel.

  Stellt den Epoch-Zeitstempel für das Ende des Verdichtungsauftrags dar.

- `metrics` – Ein RunMetrics-Objekt.

  Ein `RunMetrics`-Objekt, das Metriken für die Optimiererausführung enthält.

  Dieses Mitglied ist als veraltet gekennzeichnet. Sehen Sie sich die einzelnen Kennzahlen für Komprimierung, Aufbewahrung und Löschung verwaister Dateien an.

- `error` – UTF-8-Zeichenfolge.

  Ein Fehler, der während Optimiererausführung aufgetreten ist.

- `compactionMetrics` – Ein CompactionMetrics-Objekt.

Ein `CompactionMetrics`-Objekt, das Metriken für die Optimiererausführung enthält.

- `compactionStrategy` – UTF-8-Zeichenfolge (zulässige Werte: `binpack="BINPACK"` | `sort="SORT"` | `z-order="ZORDER"`).

  Die für den Verdichtungslauf verwendete Strategie. Gibt an, welcher Algorithmus angewendet wurde, um zu bestimmen, wie Dateien während des Komprimierungsvorgangs ausgewählt und kombiniert wurden. Gültige Werte für sind:

  - `binpack`: Kombiniert kleine Dateien zu größeren Dateien und zielt in der Regel auf Größen über 100 MB ab, wobei alle ausstehenden Löschungen angewendet werden. Dies ist die empfohlene Komprimierungsstrategie für die meisten Anwendungsfälle.
  - `sort`: Organisiert Daten auf der Grundlage bestimmter Spalten, die während der Komprimierung hierarchisch sortiert werden, wodurch die Abfrageleistung für gefilterte Operationen verbessert wird. Diese Strategie wird empfohlen, wenn Ihre Abfragen häufig nach bestimmten Spalten filtern. Um diese Strategie zu verwenden, müssen Sie zunächst mithilfe der Tabelleneigenschaft eine Sortierreihenfolge in Ihren `sort_order` Iceberg-Tabelleneigenschaften definieren.
  - `z-order`: Optimiert die Datenorganisation, indem mehrere Attribute zu einem einzigen Skalarwert zusammengefasst werden, der für die Sortierung verwendet werden kann, wodurch effiziente Abfragen über mehrere Dimensionen hinweg ermöglicht werden. Diese Strategie wird empfohlen, wenn Sie Daten über mehrere Dimensionen gleichzeitig abfragen müssen. Um diese Strategie zu verwenden, müssen Sie zunächst mithilfe der Tabelleneigenschaft eine Sortierreihenfolge in Ihren `sort_order` Iceberg-Tabelleneigenschaften definieren.
- `retentionMetrics` – Ein RetentionMetrics-Objekt.

  Ein `RetentionMetrics`-Objekt, das Metriken für die Optimiererausführung enthält.
- `orphanFileDeletionMetrics` – Ein OrphanFileDeletionMetrics-Objekt.

  Ein `OrphanFileDeletionMetrics` Objekt, das Metriken für den Optimiererlauf enthält.

BatchGetTableOptimizerEntry Struktur

Stellt einen Tabellenoptimierer dar, der während der `BatchGetTableOptimizer`-Operation abgerufen werden soll.

Felder

- `catalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.

- `databaseName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.

- `tableName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Tabelle.

- `type` – UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.

BatchTableOptimizer Struktur

Enthält Details für einen der Tabellenoptimierer, die von der `BatchGetTableOptimizer`-Operation zurückgegeben wurden.

Felder

- `catalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.

- `databaseName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.

- `tableName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Tabelle.

- `tableOptimizer` – Ein TableOptimizer-Objekt.

  Ein `TableOptimizer`-Objekt, das Details zur Konfiguration und zur letzten Ausführung eines Tabellenoptimierers enthält.

BatchGetTableOptimizerError Struktur

Enthält Details zu einem der Fehler in der von der `BatchGetTableOptimizer`-Operation zurückgegebenen Fehlerliste.

Felder

- `error` – Ein ErrorDetail-Objekt.

  Ein `ErrorDetail`-Objekt, das Code- und Meldungsdetails zu dem Fehler enthält.
- `catalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `databaseName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `tableName` – UTF-8-Zeichenfolge, mindestens 1 Byte lang.

  Der Name der Tabelle.
- `type` – UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.

RetentionConfiguration Struktur

Die Konfiguration für einen Snapshot-Aufbewahrungsoptimierer.

Felder

- `icebergConfiguration` – Ein IcebergRetentionConfiguration-Objekt.

  Die Konfiguration für einen Iceberg-Snapshot-Aufbewahrungsoptimierer.

IcebergRetentionConfiguration Struktur

Die Konfiguration für einen Iceberg Snapshot Retention Optimizer.

Felder

- `snapshotRetentionPeriodInDays` – Zahl (Ganzzahl).

  Die Anzahl der Tage, für die die Iceberg-Snapshots aufbewahrt werden sollen. Wenn keine Eingabe erfolgt, wird das entsprechende Iceberg-Tabellenkonfigurationsfeld verwendet. Falls es nicht vorhanden ist, wird der Standardwert 5 verwendet.

- `numberOfSnapshotsToRetain` – Zahl (Ganzzahl).

  Die Anzahl der Iceberg-Snapshots, die innerhalb des Aufbewahrungszeitraums aufbewahrt werden sollen. Wenn keine Eingabe erfolgt, wird das entsprechende Iceberg-Tabellenkonfigurationsfeld verwendet. Falls es nicht vorhanden ist, wird der Standardwert 1 verwendet.

- `cleanExpiredFiles` – Boolesch.

  Wenn dieser Wert auf false gesetzt ist, werden Schnappschüsse nur aus den Tabellenmetadaten gelöscht, und die zugrunde liegenden Daten und Metadatendateien werden nicht gelöscht.

OrphanFileDeletionConfiguration Struktur

Die Konfiguration für einen Optimierer für das Löschen verwaister Dateien.

Felder

- `icebergConfiguration` – Ein IcebergOrphanFileDeletionConfiguration-Objekt.

  Die Konfiguration für einen Iceberg-Optimierer für das Löschen verwaister Dateien.

IcebergOrphanFileDeletionConfiguration Struktur

Die Konfiguration für einen Iceberg-Optimierer für das Löschen verwaister Dateien.

Felder

- `orphanFileRetentionPeriodInDays` – Zahl (Ganzzahl).

  Die Anzahl der Tage, für die verwaiste Dateien aufbewahrt werden sollen, bevor sie gelöscht werden. Wenn keine Eingabe erfolgt, wird der Standardwert 3 verwendet.

- `location` – UTF-8-Zeichenfolge.

Gibt ein Verzeichnis an, in dem nach Dateien gesucht werden soll (standardmäßig der Speicherort der Tabelle). Sie können ein Unterverzeichnis anstelle des Speicherorts der Tabelle auf oberster Ebene wählen.

CompactionMetrics Struktur

Eine Struktur, die Verdichtungsmetriken für den Optimiererlauf enthält.

Felder

- `IcebergMetrics` – Ein IcebergCompactionMetrics-Objekt.

  Eine Struktur, die die Iceberg-Verdichtungsmetriken für den Optimierer-Lauf enthält.

RetentionMetrics Struktur

Eine Struktur, die Aufbewahrungsmetriken für den Optimiererlauf enthält.

Felder

- `IcebergMetrics` – Ein IcebergRetentionMetrics-Objekt.

  Eine Struktur, die die Iceberg-Aufbewahrungsmetriken für den Optimierer-Lauf enthält.

OrphanFileDeletionMetrics Struktur

Eine Struktur, die Metriken zum Löschen verwaister Dateien für den Optimierer-Lauf enthält.

Felder

- `IcebergMetrics` – Ein IcebergOrphanFileDeletionMetrics-Objekt.

  Eine Struktur, die die Iceberg-Metriken zum Löschen verwaister Dateien für den Optimierer-Lauf enthält.

IcebergCompactionMetrics Struktur

Verdichtungsmetriken für Iceberg für den Optimierer-Lauf.

Felder

- `DpuHours` – Nummer (doppelt).

  Die Anzahl der durch den Auftrag verbrauchten DPU-Stunden.
- `NumberOfDpus`— Zahl (Ganzzahl).

  Die Anzahl der vom Job DPUs verbrauchten Daten, aufgerundet auf die nächste ganze Zahl.
- `JobDurationInHour` – Nummer (doppelt).

  Die Dauer des Auftrags in Stunden.

IcebergRetentionMetrics Struktur

Kennzahlen zur Aufbewahrung von Snapshots für Iceberg für den Optimizer-Lauf.

Felder

- `DpuHours` – Nummer (doppelt).

  Die Anzahl der durch den Auftrag verbrauchten DPU-Stunden.
- `NumberOfDpus`— Zahl (Ganzzahl).

  Die Anzahl der vom Job DPUs verbrauchten Daten, aufgerundet auf die nächste ganze Zahl.
- `JobDurationInHour` – Nummer (doppelt).

  Die Dauer des Auftrags in Stunden.

IcebergOrphanFileDeletionMetrics Struktur

Metriken zum Löschen verwaister Dateien für Iceberg für den Optimizer-Lauf.

Felder

- `DpuHours` – Nummer (doppelt).

  Die Anzahl der durch den Auftrag verbrauchten DPU-Stunden.
- `NumberOfDpus`— Zahl (Ganzzahl).

  Die Anzahl der vom Job DPUs verbrauchten Daten, aufgerundet auf die nächste ganze Zahl.

- `JobDurationInHour` – Nummer (doppelt).

  Die Dauer des Auftrags in Stunden.

RunMetrics Struktur

Metriken für die Optimiererausführung.

Diese Struktur ist veraltet. Sehen Sie sich die einzelnen Metrikelemente für Komprimierung, Aufbewahrung und Löschung verwaister Dateien an.

Felder

- `NumberOfBytesCompacted` – UTF-8-Zeichenfolge.

  Die Anzahl der Bytes, die durch den Verdichtungsauftrag entfernt wurden.
- `NumberOfFilesCompacted` – UTF-8-Zeichenfolge.

  Die Anzahl der Dateien, die durch den Verdichtungsauftrag entfernt wurden.
- `NumberOfDpus` – UTF-8-Zeichenfolge.

  Die Anzahl der vom Job DPUs verbrauchten Dateien, aufgerundet auf die nächste ganze Zahl.
- `JobDurationInHour` – UTF-8-Zeichenfolge.

  Die Dauer des Auftrags in Stunden.

Operationen

- GetTableOptimizer Aktion (Python: get_table_optimizer)
- BatchGetTableOptimizer Aktion (Python: batch_get_table_optimizer)
- ListTableOptimizerRuns Aktion (Python: list_table_optimizer_runs)
- CreateTableOptimizer Aktion (Python: create_table_optimizer)
- DeleteTableOptimizer Aktion (Python: delete_table_optimizer)
- UpdateTableOptimizer Aktion (Python: update_table_optimizer)

GetTableOptimizer Aktion (Python: get_table_optimizer)

Gibt die Konfiguration aller Optimierer zurück, die einer angegebenen Tabelle zugeordnet sind.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.

Antwort

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `TableOptimizer` – Ein TableOptimizer-Objekt.

Der Optimierer, der der angegebenen Tabelle zugeordnet ist.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`
- `ThrottlingException`

BatchGetTableOptimizer Aktion (Python: batch_get_table_optimizer)

Gibt die Konfiguration für die angegebenen Tabellenoptimierer zurück.

Anforderung

- `Entries` – Erforderlich: Ein Array mit BatchGetTableOptimizerEntry-Objekten.

  Eine Liste von `BatchGetTableOptimizerEntry`-Objekten, die die abzurufenden Tabellenoptimierer angeben.

Antwort

- `TableOptimizers` – Ein Array mit BatchTableOptimizer-Objekten.

  Eine Liste von `BatchTableOptimizer`-Objekten.
- `Failures` – Ein Array mit BatchGetTableOptimizerError-Objekten.

  Eine Liste der Fehler aus der Operation.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`

- `InternalServiceException`
- `ThrottlingException`

ListTableOptimizerRuns Aktion (Python: list_table_optimizer_runs)

Listet den Verlauf früherer Optimizer-Ausführungen für eine bestimmte Tabelle auf.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.
- `MaxResults` – Zahl (Ganzzahl).

  Die maximale Anzahl von Optimiererausführungen die bei jedem Aufruf zurückgegeben werden.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `CatalogId` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.

- `DatabaseName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Liste mit Optimiererausführungen. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.
- `TableOptimizerRuns` – Ein Array mit TableOptimizerRun-Objekten.

  Eine Liste der Optimiererausführungen, die mit einer Tabelle verknüpft sind.

Fehler

- `EntityNotFoundException`
- `AccessDeniedException`
- `InvalidInputException`
- `ValidationException`
- `InternalServiceException`
- `ThrottlingException`

CreateTableOptimizer Aktion (Python: create_table_optimizer)

Erzeugt einen neuen Tabellenoptimierer für eine bestimmte Funktion.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.

- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.

- `TableOptimizerConfiguration` – Erforderlich: Ein TableOptimizerConfiguration-Objekt.

  Ein `TableOptimizerConfiguration`-Objekt, das die Konfiguration eines Tabellenoptimierers darstellt.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `ValidationException`
- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `InternalServiceException`
- `ThrottlingException`

DeleteTableOptimizer Aktion (Python: delete_table_optimizer)

Löscht einen Optimierer und alle zugehörigen Metadaten für eine Tabelle. Die Optimierung wird nicht mehr an der Tabelle durchgeführt.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`
- `ThrottlingException`

UpdateTableOptimizer Aktion (Python: update_table_optimizer)

Aktualisiert die Konfiguration für einen vorhandenen Tabellenoptimierer.

Anforderung

- `CatalogId` – Erforderlich: Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID der Tabelle.
- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank in dem Katalog, in dem sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Der Typ des Tabellenoptimierers.
- `TableOptimizerConfiguration` – Erforderlich: Ein TableOptimizerConfiguration-Objekt.

  Ein `TableOptimizerConfiguration`-Objekt, das die Konfiguration eines Tabellenoptimierers darstellt.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `ValidationException`
- `InternalServiceException`
- `ThrottlingException`
- `ConcurrentModificationException`

Crawler- und Classifier-API

Die Crawler- und Classifiers-API beschreibt die AWS Glue Crawler- und Classifier-Datentypen und umfasst die API zum Erstellen, Löschen, Aktualisieren und Auflisten von Crawlern oder Classifiern.

Themen

- Classifier-API
- Crawler-API
- API für Spaltenstatistiken
- Crawler-Scheduler-API

Classifier-API

Die Classifier-API beschreibt AWS Glue Classifier-Datentypen und umfasst die API zum Erstellen, Löschen, Aktualisieren und Auflisten von Classifiern.

Datentypen

- Classifier-Struktur
- GrokClassifier Struktur
- XMLClassifier Struktur
- JsonClassifier Struktur
- CsvClassifier Struktur
- CreateGrokClassifierRequest Struktur
- UpdateGrokClassifierRequest Struktur
- Anforderungsstruktur erstellen XMLClassifier
- Struktur der XMLClassifier Anfrage aktualisieren
- CreateJsonClassifierRequest Struktur
- UpdateJsonClassifierRequest Struktur
- CreateCsvClassifierRequest Struktur
- UpdateCsvClassifierRequest Struktur

Classifier-Struktur

Classifier werden während einer Durchsuchungsaufgabe ausgelöst. Ein Classifier prüft, ob eine bestimmte Datei ein Format hat, mit dem er umgehen kann. Wenn dies der Fall ist, erstellt der Classifier ein Schema in Form eines `StructType`-Objekts, das dem Datenformat entspricht.

Sie können die Standardklassifikatoren verwenden, die diese AWS Glue zur Verfügung stellen, oder Sie können Ihre eigenen Klassifikatoren schreiben, um Ihre Datenquellen bestmöglich zu kategorisieren und die entsprechenden Schemas zu spezifizieren, die für sie verwendet werden sollen. Ein Classifier kann ein `grok`-Classifier, ein `XML`-Classifier oder ein `JSON`-Classifier oder ein benutzerdefinierter `CSV`-Classifier sein, wie in einem der Felder im `Classifier`-Objekt angegeben.

Felder

- `GrokClassifier` – Ein GrokClassifier-Objekt.

  Ein Classifier, der `grok` verwendet.
- `XMLClassifier` – Ein XMLClassifier-Objekt.

  Ein Classifier für XML-Inhalte.
- `JsonClassifier` – Ein JsonClassifier-Objekt.

  Ein Classifier für JSON-Inhalte.
- `CsvClassifier` – Ein CsvClassifier-Objekt.

  Ein Classifier für durch Kommata getrennte Werte (CSV).

GrokClassifier Struktur

Ein Classifier, der `grok`-Muster verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.
- `Classification` – Erforderlich: UTF-8-Zeichenfolge.

Eine ID des Datenformates, das der Classifier abgleicht, beispielsweise Protokolle von Twitter, JSON oder Omniture usw.

- `CreationTime` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier registriert wurde.

- `LastUpdated` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier zuletzt aktualisiert wurde.

- `Version` – Zahl (lang).

  Die Version dieses Classifiers.

- `GrokPattern` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum A Logstash Grok string pattern.

  Das Grok-Muster, das von diesem Classifier auf einen Datenspeicher angewendet wird. Weitere Informationen finden Sie in den integrierten Mustern unter Schreiben von benutzerdefinierten Classifiern.

- `CustomPatterns` – UTF-8-Zeichenfolge, nicht mehr als 16000 Bytes lang, passend zum URI address multi-line string pattern.

  Optionale benutzerdefinierte Grok-Muster, die von diesem Classifier definiert werden. Weitere Informationen finden Sie in den benutzerdefinierten Mustern unter Schreiben von benutzerdefinierten Classifiern.

XMLClassifier Struktur

Ein Classifier für `XML`-Inhalte.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `Classification` – Erforderlich: UTF-8-Zeichenfolge.

  Eine ID des Datenformats, dass mit dem Classifier übereinstimmt.

- `CreationTime` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier registriert wurde.

- `LastUpdated` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier zuletzt aktualisiert wurde.

- `Version` – Zahl (lang).

  Die Version dieses Classifiers.

- `RowTag` – UTF-8-Zeichenfolge.

  Das XML-Tag, das das Element festlegt, das jeden Datensatz in einem XML-Dokument enthält, das analysiert wird. Damit kann kein selbstschließendes Element (geschlossen von `/>`) identifiziert werden. Ein leeres Zeilenelement, das ausschließlich Attribute enthält, kann analysiert werden, solange es mit einem schließenden Tag endet (z. B. ist `<row item_a="A" item_b="B"></row>` in Ordnung, `<row item_a="A" item_b="B" />` aber nicht).

JsonClassifier Struktur

Ein Classifier für `JSON`-Inhalte.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `CreationTime` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier registriert wurde.

- `LastUpdated` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier zuletzt aktualisiert wurde.

- `Version` – Zahl (lang).

  Die Version dieses Classifiers.

- `JsonPath` – Erforderlich: UTF-8-Zeichenfolge.

Eine `JsonPath` Zeichenfolge, die die JSON-Daten für den zu klassifizierenden Klassifikator definiert. AWS Glue unterstützt eine Teilmenge von JsonPath, wie unter Benutzerdefinierte Klassifikatoren schreiben JsonPath beschrieben.

CsvClassifier Struktur

Ein Classifier für benutzerdefinierte `CSV`-Inhalte.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.
- `CreationTime` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier registriert wurde.
- `LastUpdated` – Zeitstempel.

  Der Zeitpunkt, an dem dieser Classifier zuletzt aktualisiert wurde.
- `Version` – Zahl (lang).

  Die Version dieses Classifiers.
- `Delimiter` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Eine benutzerdefiniertes Symbol zur Bezeichnung, wodurch die einzelnen Spalteneinträge in der Zeile voneinander getrennt werden.
- `QuoteSymbol` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Ein benutzerdefiniertes Symbol zur Bezeichnung, wodurch Inhalte zu einem einzelnen Spaltenwert miteinander kombiniert werden. Muss sich von dem Spaltentrennzeichen unterscheiden.
- `ContainsHeader` – UTF-8-Zeichenfolge (zulässige Werte: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Gibt an, ob die CSV-Datei eine Kopfzeile enthält.
- `Header` – Ein UTF-8-Zeichenfolgen-Array.

Eine Liste von Zeichenfolgen, durch die Spaltennamen darstellt werden.

- `DisableValueTrimming` – Boolesch.

  Gibt an, dass Werte vor dem Identifizieren des Typs der Spaltenwerte nicht abgetrennt werden sollen. Der Standardwert ist `true`.

- `AllowSingleColumn` – Boolesch.

  Aktiviert die Verarbeitung von Dateien, die nur eine Spalte enthalten.

- `CustomDatatypeConfigured` – Boolesch.

  Ermöglicht die Konfiguration des benutzerdefinierten Datentyps.

- `CustomDatatypes` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von benutzerdefinierten Datentypen, einschließlich „BINARY“, „BOOLEAN“, „DATE“, „DECIMAL“, „DOUBLE“, „FLOAT“, „INT“, „LONG“, „SHORT“, „STRING“, „TIMESTAMP“.

- `Serde` – UTF-8-Zeichenfolge (zulässige Werte: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Legt die SerDe für die Verarbeitung erforderliche CSV im Klassifikator fest, die im Datenkatalog angewendet wird. Gültige Werte sind `OpenCSVSerDe`, `LazySimpleSerDe` und `None`. Sie können den `None`-Wert angeben, wenn der Crawler die Erkennung durchführen soll.

CreateGrokClassifierRequest Struktur

Gibt einen zu erstellenden `grok`-Classifier für `CreateClassifier` an.

Felder

- `Classification` – Erforderlich: UTF-8-Zeichenfolge.

  Eine Kennung des Datenformats, dem der Classifier entspricht, z. B. Twitter, JSON, Omniture-Logs, Amazon CloudWatch Logs usw.

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des neuen Classifiers.

- `GrokPattern` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum A Logstash Grok string pattern.

Das Grok-Muster, das von diesem Classifier verwendet wird.

- `CustomPatterns` – UTF-8-Zeichenfolge, nicht mehr als 16000 Bytes lang, passend zum URI address multi-line string pattern.

  Optionale benutzerdefinierte Grok-Muster, die von diesem Classifier verwendet werden.

UpdateGrokClassifierRequest Struktur

Gibt einen Grok-Classifier an, der bei der Weiterleitung an `UpdateClassifier` aktualisiert werden soll.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `GrokClassifier`.

- `Classification` – UTF-8-Zeichenfolge.

  Eine Kennung des Datenformats, dem der Classifier entspricht, z. B. Twitter, JSON, Omniture-Logs, Amazon CloudWatch Logs usw.

- `GrokPattern` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Bytes lang, passend zum A Logstash Grok string pattern.

  Das Grok-Muster, das von diesem Classifier verwendet wird.

- `CustomPatterns` – UTF-8-Zeichenfolge, nicht mehr als 16000 Bytes lang, passend zum URI address multi-line string pattern.

  Optionale benutzerdefinierte Grok-Muster, die von diesem Classifier verwendet werden.

Anforderungsstruktur erstellen XMLClassifier

Gibt einen zu erstellenden XML-Classifier für `CreateClassifier` an.

Felder

- `Classification` – Erforderlich: UTF-8-Zeichenfolge.

Eine ID des Datenformats, dass mit dem Classifier übereinstimmt.

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `RowTag` – UTF-8-Zeichenfolge.

  Das XML-Tag, das das Element festlegt, das jeden Datensatz in einem XML-Dokument enthält, das analysiert wird. Damit kann kein selbstschließendes Element (geschlossen von `/>`) identifiziert werden. Ein leeres Zeilenelement, das ausschließlich Attribute enthält, kann analysiert werden, solange es mit einem schließenden Tag endet (z. B. ist `<row item_a="A" item_b="B"></row>` in Ordnung, `<row item_a="A" item_b="B" />` aber nicht).

Struktur der XMLClassifier Anfrage aktualisieren

Gibt einen zu aktualisierenden XML-Classifier an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `Classification` – UTF-8-Zeichenfolge.

  Eine ID des Datenformats, dass mit dem Classifier übereinstimmt.

- `RowTag` – UTF-8-Zeichenfolge.

  Das XML-Tag, das das Element festlegt, das jeden Datensatz in einem XML-Dokument enthält, das analysiert wird. Beachten Sie, dass dies kein selbstschließendes Element identifizieren kann (geschlossen von `/>`). Ein leeres Zeilenelement, das ausschließlich Attribute enthält, kann analysiert werden, solange es mit einem schließenden Tag endet (z. B. ist `<row item_a="A" item_b="B"></row>` in Ordnung, `<row item_a="A" item_b="B" />` aber nicht).

CreateJsonClassifierRequest Struktur

Gibt einen zu erstellenden JSON-Classifier für `CreateClassifier` an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `JsonPath` – Erforderlich: UTF-8-Zeichenfolge.

  Eine `JsonPath` Zeichenfolge, die die JSON-Daten für den zu klassifizierenden Klassifikator definiert. AWS Glue unterstützt eine Teilmenge von JsonPath, wie unter Benutzerdefinierte Klassifikatoren schreiben JsonPath beschrieben.

UpdateJsonClassifierRequest Struktur

Gibt einen zu aktualisierenden JSON-Classifier an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `JsonPath` – UTF-8-Zeichenfolge.

  Eine `JsonPath` Zeichenfolge, die die JSON-Daten für den zu klassifizierenden Klassifikator definiert. AWS Glue unterstützt eine Teilmenge von JsonPath, wie unter Benutzerdefinierte Klassifikatoren schreiben JsonPath beschrieben.

CreateCsvClassifierRequest Struktur

Gibt einen benutzerdefinierten CSV-Classifier für `CreateClassifier` an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `Delimiter` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Eine benutzerdefiniertes Symbol zur Bezeichnung, wodurch die einzelnen Spalteneinträge in der Zeile voneinander getrennt werden.
- `QuoteSymbol` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Ein benutzerdefiniertes Symbol zur Bezeichnung, wodurch Inhalte zu einem einzelnen Spaltenwert miteinander kombiniert werden. Muss sich von dem Spaltentrennzeichen unterscheiden.
- `ContainsHeader` – UTF-8-Zeichenfolge (zulässige Werte: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Gibt an, ob die CSV-Datei eine Kopfzeile enthält.
- `Header` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Zeichenfolgen, durch die Spaltennamen darstellt werden.
- `DisableValueTrimming` – Boolesch.

  Gibt an, dass Werte vor dem Identifizieren des Typs der Spaltenwerte nicht abgetrennt werden sollen. Der Standardwert ist "True".
- `AllowSingleColumn` – Boolesch.

  Aktiviert die Verarbeitung von Dateien, die nur eine Spalte enthalten.
- `CustomDatatypeConfigured` – Boolesch.

  Ermöglicht die Konfiguration benutzerdefinierter Datentypen.
- `CustomDatatypes` – Ein UTF-8-Zeichenfolgen-Array.

  Erzeugt eine Liste der unterstützten benutzerdefinierten Datentypen.
- `Serde` – UTF-8-Zeichenfolge (zulässige Werte: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Legt die SerDe für die Verarbeitung erforderliche CSV im Klassifikator fest, die im Datenkatalog angewendet wird. Gültige Werte sind `OpenCSVSerDe`, `LazySimpleSerDe` und `None`. Sie können den `None`-Wert angeben, wenn der Crawler die Erkennung durchführen soll.

UpdateCsvClassifierRequest Struktur

Gibt einen benutzerdefinierten CSV-Classifier an, der aktualisiert werden soll.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Classifiers.

- `Delimiter` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Eine benutzerdefiniertes Symbol zur Bezeichnung, wodurch die einzelnen Spalteneinträge in der Zeile voneinander getrennt werden.

- `QuoteSymbol` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1 Byte lang, passend zum Custom string pattern #26.

  Ein benutzerdefiniertes Symbol zur Bezeichnung, wodurch Inhalte zu einem einzelnen Spaltenwert miteinander kombiniert werden. Muss sich von dem Spaltentrennzeichen unterscheiden.

- `ContainsHeader` – UTF-8-Zeichenfolge (zulässige Werte: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Gibt an, ob die CSV-Datei eine Kopfzeile enthält.

- `Header` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Zeichenfolgen, durch die Spaltennamen darstellt werden.

- `DisableValueTrimming` – Boolesch.

  Gibt an, dass Werte vor dem Identifizieren des Typs der Spaltenwerte nicht abgetrennt werden sollen. Der Standardwert ist "True".

- `AllowSingleColumn` – Boolesch.

  Aktiviert die Verarbeitung von Dateien, die nur eine Spalte enthalten.

- `CustomDatatypeConfigured` – Boolesch.

  Gibt die Konfiguration benutzerdefinierter Datentypen an.

- `CustomDatatypes` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt eine Liste der unterstützten benutzerdefinierten Datentypen an.

- `Serde` – UTF-8-Zeichenfolge (zulässige Werte: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

Legt die SerDe für die Verarbeitung erforderliche CSV im Klassifikator fest, die im Datenkatalog angewendet wird. Gültige Werte sind `OpenCSVSerDe`, `LazySimpleSerDe` und `None`. Sie können den `None`-Wert angeben, wenn der Crawler die Erkennung durchführen soll.

Operationen

- CreateClassifier Aktion (Python: create_classifier)
- DeleteClassifier Aktion (Python: delete_classifier)
- GetClassifier Aktion (Python: get_classifier)
- GetClassifiers Aktion (Python: get_classifiers)
- UpdateClassifier Aktion (Python: update_classifier)

CreateClassifier Aktion (Python: create_classifier)

Erstellt einen Classifier im Konto des Benutzers. Dies kann ein `GrokClassifier`, ein `XMLClassifier`, ein `JsonClassifier` oder ein `CsvClassifier` sein, je nachdem, welches Feld der Anforderung vorhanden ist.

Anforderung

- `GrokClassifier` – Ein CreateGrokClassifierRequest-Objekt.

  Ein `GrokClassifier`-Objekt, das den zu erstellenden Classifier angibt.
- `XMLClassifier` – Ein XMLClassifierAnfrage erstellen-Objekt.

  Ein `XMLClassifier`-Objekt, das den zu erstellenden Classifier angibt.
- `JsonClassifier` – Ein CreateJsonClassifierRequest-Objekt.

  Ein `JsonClassifier`-Objekt, das den zu erstellenden Classifier angibt.
- `CsvClassifier` – Ein CreateCsvClassifierRequest-Objekt.

  Ein `CsvClassifier`-Objekt, das den zu erstellenden Classifier angibt.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`

DeleteClassifier Aktion (Python: delete_classifier)

Entfernt einen Classifier aus dem Data Catalog.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des zu entfernenden Classifiers.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifier Aktion (Python: get_classifier)

Abrufen eines Classifiers nach Namen.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des abzurufenden Classifiers.

Antwort

- `Classifier` – Ein Classifier-Objekt.

  Der angeforderte Classifier.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifiers Aktion (Python: get_classifiers)

Listet alle Classifier-Objekte im Data Catalog auf.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Größe der zurückzugebenden Liste (optional).
- `NextToken` – UTF-8-Zeichenfolge.

  Ein optionales Fortsetzungs-Token.

Antwort

- `Classifiers` – Ein Array mit Classifier-Objekten.

  Die angeforderte Liste der Classifier-Objekte.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token.

Fehler

- `OperationTimeoutException`

UpdateClassifier Aktion (Python: update_classifier)

Ändert einen vorhandenen Classifier (einen `GrokClassifier`, einen `XMLClassifier`, einen `JsonClassifier` oder einen `CsvClassifier`, je nachdem, welches Feld vorhanden ist).

Anforderung

- `GrokClassifier` – Ein UpdateGrokClassifierRequest-Objekt.

  Ein `GrokClassifier`-Objekt mit aktualisierten Feldern.
- `XMLClassifier` – Ein XMLClassifierAnfrage aktualisieren-Objekt.

  Ein `XMLClassifier`-Objekt mit aktualisierten Feldern.
- `JsonClassifier` – Ein UpdateJsonClassifierRequest-Objekt.

  Ein `JsonClassifier`-Objekt mit aktualisierten Feldern.
- `CsvClassifier` – Ein UpdateCsvClassifierRequest-Objekt.

  Ein `CsvClassifier`-Objekt mit aktualisierten Feldern.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `VersionMismatchException`
- `EntityNotFoundException`
- `OperationTimeoutException`

Crawler-API

Die Crawler-API beschreibt AWS Glue Crawler-Datentypen sowie die API zum Erstellen, Löschen, Aktualisieren und Auflisten von Crawlern.

Datentypen

- Crawler-Struktur
- Planstruktur
- CrawlerTargets Struktur
- S3Target-Struktur
- DeltaCatalogTarget S3-Struktur
- DeltaDirectTarget S3-Struktur
- JdbcTarget Struktur
- DBTarget Mongo-Struktur
- DBTarget Dynamo-Struktur
- DeltaTarget Struktur
- IcebergTarget Struktur
- HudiTarget Struktur
- CatalogTarget Struktur
- CrawlerMetrics Struktur
- CrawlerHistory Struktur
- CrawlsFilter Struktur
- SchemaChangePolicy Struktur
- LastCrawlInfo Struktur
- RecrawlPolicy Struktur
- LineageConfiguration Struktur
- LakeFormationConfiguration Struktur

Crawler-Struktur

Gibt ein Crawler-Programm an, das eine Datenquelle untersucht und Classifier verwendet, um deren Schema zu ermitteln. Bei Erfolg erfasst der Crawler Metadaten über die Datenquelle im AWS Glue Data Catalog.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Crawlers.

- `Role` – UTF-8-Zeichenfolge.

  Der Amazon-Ressourcenname (ARN) einer IAM-Rolle, die für den Zugriff auf Kundenressourcen verwendet wird, wie z. B. Amazon Simple Storage Service (Amazon S3)-Daten.

- `Targets` – Ein CrawlerTargets-Objekt.

  Eine Sammlung von Zielen zum Crawlen.

- `DatabaseName` – UTF-8-Zeichenfolge.

  Der Name der Datenbank, in der die Crawler-Ausgabe gespeichert wird.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Crawlers.

- `Classifiers` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der UTF-8-Zeichenfolgen, die die benutzerdefinierten Classifier des Crawlers angeben.

- `RecrawlPolicy` – Ein RecrawlPolicy-Objekt.

  Eine Richtlinie, die angibt, ob das Crawling für den gesamten Datensatz erneut ausgeführt werden soll oder nur Ordner gecrawlt werden sollen, die seit der letzten Crawler-Ausführung hinzugefügt wurden.

- `SchemaChangePolicy` – Ein SchemaChangePolicy-Objekt.

  Die Richtlinie, in der die Aktualisierungs- und Löschverhaltensweisen für den Crawler festgelegt sind.

- `LineageConfiguration` – Ein LineageConfiguration-Objekt.

  Eine Konfiguration, die angibt, ob die Datenherkunft für den Crawler aktiviert ist.

- `State` – UTF-8-Zeichenfolge (zulässige Werte: `READY` | `RUNNING` | `STOPPING`).

  Gibt an, ob der Crawler ausgeführt wird oder ob eine Ausführung noch aussteht.

- `TablePrefix` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Das Präfix, das den Namen der Tabellen, die erstellt werden, hinzugefügt wird.

- `Schedule` – Ein Plan-Objekt.

  Bei geplanten Crawlern ist dies der Zeitplan, wenn der Crawler ausgeführt wird.

- `CrawlElapsedTime` – Zahl (lang).

  Wenn der Crawler ausgeführt wird, ist die gesamte Zeit enthalten, die seit Beginn des letzten Crawls verstrichen ist.

- `CreationTime` – Zeitstempel.

  Die Uhrzeit, zu der der Crawler erstellt wurde.

- `LastUpdated` – Zeitstempel.

  Die Uhrzeit, zu der der Crawler zuletzt aktualisiert wurde.

- `LastCrawl` – Ein LastCrawlInfo-Objekt.

  Der Status des letzten Crawls und möglicherweise Fehlerinformationen, wenn ein Fehler aufgetreten ist.

- `Version` – Zahl (lang).

  Die Version des Crawlers.

- `Configuration` – UTF-8-Zeichenfolge.

  Crawler-Konfigurationsinformationen. Mit dieser versionierten JSON-Zeichenfolge können Benutzer Verhaltensaspekte eines Crawlers angeben. Weitere Informationen finden Sie unter Festlegen von Crawler-Konfigurationsoptionen.

- `CrawlerSecurityConfiguration` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Der Name der `SecurityConfiguration` Struktur, die von diesem Crawler verwendet werden soll.

- `LakeFormationConfiguration` – Ein LakeFormationConfiguration-Objekt.

  Gibt an, ob der Crawler AWS Lake Formation Anmeldeinformationen für den Crawler anstelle der Anmeldeinformationen der IAM-Rolle verwenden soll.

Planstruktur

Ein Planungsobjekt, das eine `cron`-Anweisung zum Planen eines Ereignisses verwendet.

Felder

- `ScheduleExpression` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.
- `State` – UTF-8-Zeichenfolge (zulässige Werte: `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  Der Status des Zeitplans.

CrawlerTargets Struktur

Gibt Datenspeicher an, die gecrawlt werden sollen.

Felder

- `S3Targets` – Ein Array mit S3Target-Objekten.

  Gibt Amazon Simple Storage Service (Amazon S3)-Ziele an.
- `JdbcTargets` – Ein Array mit JdbcTarget-Objekten.

  Gibt JDBC-Ziele an.
- `MongoDBTargets` – Ein Array mit Mongo DBTarget-Objekten.

  Gibt Amazon-DocumentDB- oder MongoDB-Ziele an.
- `DynamoDBTargets` – Ein Array mit Dynamo DBTarget-Objekten.

  Gibt Amazon DynamoDB-Ziele an.
- `CatalogTargets` – Ein Array mit CatalogTarget-Objekten.

  Spezifiziert AWS Glue Data Catalog Ziele.
- `DeltaTargets` – Ein Array mit DeltaTarget-Objekten.

  Gibt Delta-Datenspeicherziele an.

- `IcebergTargets` – Ein Array mit IcebergTarget-Objekten.

  Gibt Apache-Iceberg-Datenspeicherziele an.

- `HudiTargets` – Ein Array mit HudiTarget-Objekten.

  Gibt Apache-Hudi-Datenspeicherziele an.

S3Target-Struktur

Gibt einen Datenspeicher in Amazon Simple Storage Service (Amazon S3) an.

Felder

- `Path` – UTF-8-Zeichenfolge.

  Der Pfad zum Amazon S3-Ziel.

- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Globmuster, die zum Ausschließen aus dem Crawl-Vorgang verwendet werden. Weitere Informationen finden Sie unter Katalogisieren von Tabellen mit einem Crawler.

- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name einer Verbindung, die es einer Aufgabe oder einem Crawler ermöglicht, innerhalb einer Amazon Virtual Private Cloud-Umgebung (Amazon VPC) auf Daten in Amazon S3 zuzugreifen.

- `SampleSize` – Zahl (Ganzzahl).

  Legt die Anzahl der Dateien in jedem Ordner fest, die beim Crawling von Beispieldateien in einem Datensatz durchsucht werden sollen. Wenn nicht festgelegt, werden alle Dateien durchsucht. Ein gültiger Wert ist eine ganze Zahl zwischen 1 und 249.

- `EventQueueArn` – UTF-8-Zeichenfolge.

  Ein gültiger Amazon SQS ARN. Beispiel, `arn:aws:sqs:region:account:sqs`.

- `DlqEventQueueArn` – UTF-8-Zeichenfolge.

  Ein gültiger Amazon Dead Letter SQS ARN. Beispiel, `arn:aws:sqs:region:account:deadLetterQueue`.

DeltaCatalogTarget S3-Struktur

Gibt ein Ziel an, das in eine Delta Lake-Datenquelle im AWS Glue Datenkatalog schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.
- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.
- `SchemaChangePolicy` – Ein CatalogSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

DeltaDirectTarget S3-Struktur

Gibt ein Ziel an, das in eine Delta Lake-Datenquelle in schreibt Amazon S3.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.

- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Amazon-S3-Pfad Ihrer Delta-Lake-Datenquelle, in die geschrieben werden soll.

- `Compression` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).

- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Gibt die Anzahl der Zielpartitionen für die Verteilung von Delta Lake-Datensatzdateien in Amazon S3 an.

- `Format`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `json="JSON"` | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | | `delta="DELTA"` `iceberg="ICEBERG"` `hyper="HYPER"` |`xml="XML"`).

  Gibt das Datenausgabeformat für das Ziel an.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.

- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

JdbcTarget Struktur

Gibt einen JDBC-Datenspeicher an, der gecrawlt werden sollen.

Felder

- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung, die für das JDBC-Ziel verwendet werden soll.
- `Path` – UTF-8-Zeichenfolge.

  Der Pfad des JDBC-Ziels.
- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Globmuster, die zum Ausschließen aus dem Crawl-Vorgang verwendet werden. Weitere Informationen finden Sie unter Katalogisieren von Tabellen mit einem Crawler.
- `EnableAdditionalMetadata` – Ein UTF-8-Zeichenfolgen-Array.

  Geben Sie einen Wert von `RAWTYPES` oder `COMMENTS` an, um zusätzliche Metadaten in Tabellenantworten zu aktivieren. `RAWTYPES` stellt den Datentyp auf nativer Ebene bereit. `COMMENTS` stellt Kommentare bereit, die einer Spalte oder Tabelle in der Datenbank zugeordnet sind.

  Wenn Sie keine zusätzlichen Metadaten benötigen, lassen Sie das Feld leer.

DBTarget Mongo-Struktur

Gibt einen Amazon-DocumentDB- oder MongoDB-Datastore für das Crawling an.

Felder

- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung, die für das Amazon-DocumentDB- oder MongoDB-Ziel verwendet werden soll.
- `Path` – UTF-8-Zeichenfolge.

  Der Pfad des Amazon-DocumentDB- oder MongoDB-Ziels (Datenbank/Sammlung).
- `ScanAll` – Boolesch.

Gibt an, ob alle Datensätze gescannt oder Zeilen aus der Tabelle beispielhaft abgefragt werden sollen. Das Scannen aller Datensätze kann eine lange Zeit in Anspruch nehmen, wenn die Tabelle keinen hohen Durchsatz hat.

Der Wert eins `true` gibt an, dass alle Datensätze gescannt werden, während bei dem Wert `false` Datensätze beispielhaft abgefragt werden. Bei keiner Wertangabe wird standardmäßig der Wert `true` verwendet.

DBTarget Dynamo-Struktur

Gibt eine Amazon DynamoDB-Tabelle an, die gecrawlt werden soll.

Felder

- `Path` – UTF-8-Zeichenfolge.

  Der Name der DynamoDB-Tabelle für das Crawling.

- `scanAll` – Boolesch.

  Gibt an, ob alle Datensätze gescannt oder Zeilen aus der Tabelle beispielhaft abgefragt werden sollen. Das Scannen aller Datensätze kann eine lange Zeit in Anspruch nehmen, wenn die Tabelle keinen hohen Durchsatz hat.

  Der Wert eins `true` gibt an, dass alle Datensätze gescannt werden, während bei dem Wert `false` Datensätze beispielhaft abgefragt werden. Bei keiner Wertangabe wird standardmäßig der Wert `true` verwendet.

- `scanRate` – Nummer (doppelt).

  Der Prozentsatz der konfigurierten Lesekapazitätseinheiten, die vom AWS Glue Crawler verwendet werden sollen. Lesekapazitätseinheiten sind ein von DynamoDB definierter Begriff und ein numerischer Wert, der als Ratenbegrenzer für die Anzahl der Lesevorgänge fungiert, die pro Sekunde für diese Tabelle durchgeführt werden können.

  Die gültigen Werte sind null oder ein Wert zwischen 0,1 und 1,5. Der Nullwert wird verwendet, wenn der Benutzer keinen Wert bereitstellt, und nutzt standardmäßig 0,5 der konfigurierten Lesekapazitätseinheit (für bereitgestellte Tabellen) oder 0,25 der maximal konfigurierten Lesekapazitätseinheit (für Tabellen, die den On-Demand-Modus verwenden).

DeltaTarget Struktur

Gibt einen Delta-Datenspeicher an, um eine oder mehrere Delta-Tabellen zu crawlen.

Felder

- `DeltaTables` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Amazon-S3-Pfade zu den Delta-Tabellen.
- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung, die zum Delta-Tabellenziel hergestellt werden soll.
- `WriteManifest` – Boolesch.

  Gibt an, ob die Manifestdateien in den Delta-Tabellenpfad geschrieben werden sollen.
- `CreateNativeDeltaTable` – Boolesch.

  Gibt an, ob der Crawler native Tabellen erstellt, um die Integration mit Abfragemodulen zu ermöglichen, die die direkte Abfrage des Delta-Transaktionsprotokolls unterstützen.

IcebergTarget Struktur

Gibt eine Apache-Iceberg-Datenquelle an, in der Iceberg-Tabellen in Amazon S3 gespeichert werden.

Felder

- `Paths` – Ein UTF-8-Zeichenfolgen-Array.

  Ein oder mehrere Amazon S3 Pfade, die Iceberg-Metadatenordner als `s3://bucket/prefix` enthalten.
- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung, die für die Verbindung mit dem Iceberg-Ziel verwendet werden soll.
- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Globmuster, die zum Ausschließen aus dem Crawl-Vorgang verwendet werden. Weitere Informationen finden Sie unter Katalogisieren von Tabellen mit einem Crawler.
- `MaximumTraversalDepth` – Zahl (Ganzzahl).

Die maximale Tiefe der Amazon S3 Pfade, die der Crawler durchqueren kann, um den Iceberg-Metadatenordner in Ihrem Pfad zu finden. Amazon S3 Wird zur Begrenzung der Crawler-Laufzeit verwendet.

HudiTarget Struktur

Gibt eine Apache-Hudi-Datenquelle an.

Felder

- `Paths` – Ein UTF-8-Zeichenfolgen-Array.

  Ein Array von Amazon S3 Positionszeichenfolgen für Hudi, die jeweils den Stammordner angeben, in dem sich die Metadatendateien für eine Hudi-Tabelle befinden. Der Hudi-Ordner befindet sich möglicherweise in einem untergeordneten Ordner des Stammordners.

  Der Crawler durchsucht alle Ordner unterhalb eines Pfades nach einem Hudi-Ordner.
- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung, die zum Herstellen einer Verbindung mit dem Hudi-Ziel verwendet werden soll. Wenn Ihre Hudi-Dateien in Buckets gespeichert sind, die eine VPC-Autorisierung erfordern, können Sie deren Verbindungseigenschaften hier festlegen.
- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Globmuster, die zum Ausschließen aus dem Crawl-Vorgang verwendet werden. Weitere Informationen finden Sie unter Katalogisieren von Tabellen mit einem Crawler.
- `MaximumTraversalDepth` – Zahl (Ganzzahl).

  Die maximale Tiefe der Amazon S3 Pfade, die der Crawler durchqueren kann, um den Hudi-Metadatenordner in Ihrem Pfad zu finden. Amazon S3 Wird zur Begrenzung der Crawler-Laufzeit verwendet.

CatalogTarget Struktur

Spezifiziert ein AWS Glue Data Catalog Ziel.

Felder

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, die synchronisiert werden soll.
- `Tables` – Erforderlich:Ein Array mit UTF-8-Zeichenfolgen, mindestens 1 Zeichenfolge.

  Eine Liste der Tabellen, die synchronisiert werden sollen.
- `ConnectionName`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der Name der Verbindung für eine von Amazon S3 unterstützte Datenkatalog-Tabelle als Ziel des Crawling bei Verwendung eines `Catalog`-Verbindungstyps gepaart mit einem `NETWORK`-Verbindungstyp.
- `EventQueueArn` – UTF-8-Zeichenfolge.

  Ein gültiger Amazon SQS ARN. Beispiel, `arn:aws:sqs:region:account:sqs`.
- `DlqEventQueueArn` – UTF-8-Zeichenfolge.

  Ein gültiger Amazon Dead Letter SQS ARN. Beispiel, `arn:aws:sqs:region:account:deadLetterQueue`.

CrawlerMetrics Struktur

Metriken für einen bestimmten Crawler.

Felder

- `CrawlerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Crawlers.
- `TimeLeftSeconds` – Zahl (Double), nicht mehr als Keine.

  Die geschätzte Zeit bis zum Abschließen eines laufenden Crawls.
- `StillEstimating` – Boolesch.

  "True", wenn der Crawler immer noch schätzt, wie lange es dauert, um diesen Durchgang abzuschließen.

- `LastRuntimeSeconds` – Zahl (Double), nicht mehr als Keine.

  Die Dauer des letzten Durchgangs des Crawlers in Sekunden.
- `MedianRuntimeSeconds` – Zahl (Double), nicht mehr als Keine.

  Die mittlere Dauer der Durchgänge dieses Crawlers in Sekunden.
- `TablesCreated` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Anzahl der Tabellen, die von diesem Crawler erstellt wurden.
- `TablesUpdated` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Anzahl der Tabellen, die von diesem Crawler aktualisiert wurden.
- `TablesDeleted` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Anzahl der Tabellen, die von diesem Crawler gelöscht wurden.

CrawlerHistory Struktur

Enthält die Informationen für eine Ausführung eines Crawlers.

Felder

- `CrawlId` – UTF-8-Zeichenfolge.

  Ein UUID-Bezeichner für jedes Crawling.
- `State` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `COMPLETED` | `FAILED` | `STOPPED`).

  Der Status des Crawls.
- `StartTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Crawl gestartet wurde.
- `EndTime` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Crawl beendet wurde.
- `Summary` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Zusammenfassung der Ausführung des spezifischen Crawls in JSON. Enthält die Katalogtabellen und Partitionen, die hinzugefügt, aktualisiert oder gelöscht wurden.

- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Wenn ein Fehler aufgetreten ist, ist dies dem Crawl zugeordnet.
- `LogGroup` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log group string pattern.

  Die dem Crawl zugeordnete Protokollgruppe.
- `LogStream` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log-stream string pattern.

  Der dem Crawl zugeordnete Protokoll-Stream.
- `MessagePrefix` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Das Präfix für eine CloudWatch Nachricht über diesen Crawl.
- `DPUHour` – Zahl (Double), nicht mehr als Keine.

  Die Anzahl der Datenverarbeitungseinheiten (DPU), die in Stunden für den Crawl verwendet wurden.

CrawlsFilter Struktur

Eine Liste von Feldern, Komparatoren und Werten, die Sie zum Filtern der Crawler-Ausführungen für einen bestimmten Crawler verwenden können.

Felder

- `FieldName` – UTF-8-Zeichenfolge (zulässige Werte: `CRAWL_ID` | `STATE` | `START_TIME` | `END_TIME` | `DPU_HOUR`).

  Ein Schlüssel, der zum Filtern der Crawler-Ausführungen für einen bestimmten Crawler verwendet wird. Gültige Werte für jeden Feldnamen sind:
  - `CRAWL_ID`: Eine Zeichenfolge, die den UUID-Bezeichner für einen Crawl darstellt.
  - `STATE`: Eine Zeichenfolge, die den Status des Crawls darstellt.
  - `START_TIME` und `END_TIME`: Der Zeitstempel der Epoche in Millisekunden.
  - `DPU_HOUR`: Die Anzahl der Stunden der Datenverarbeitungseinheit (DPU), die für den Crawl verwendet wurden.

- `FilterOperator` – UTF-8-Zeichenfolge (zulässige Werte: `GT` | `GE` | `LT` | `LE` | `EQ` | `NE`).

  Ein definierter Komparator, der mit dem Wert arbeitet. Die verfügbaren Operatoren sind:

  - `GT`: Größer als.
  - `GE`: Größer als oder gleich.
  - `LT`: Weniger als.
  - `LE`: Weniger als oder gleich.
  - `EQ`: Gleich.
  - `NE`: Nicht gleich.
- `FieldValue` – UTF-8-Zeichenfolge.

  Der für den Vergleich im Crawling-Feld angegebenen Wert.

SchemaChangePolicy Struktur

Eine Richtlinie, in der die Aktualisierungs- und Löschverhaltensweisen für den Crawler festgelegt sind.

Felder

- `UpdateBehavior` – UTF-8-Zeichenfolge (zulässige Werte: `LOG` | `UPDATE_IN_DATABASE`).

  Das Aktualisierungsverhalten, wenn der Crawler ein geändertes Schema findet.
- `DeleteBehavior` – UTF-8-Zeichenfolge (zulässige Werte: `LOG` | `DELETE_FROM_DATABASE` | `DEPRECATE_IN_DATABASE`).

  Das Löschverhalten, wenn der Crawler ein gelöschtes Objekt findet.

LastCrawlInfo Struktur

Status- und Fehlerinformationen über den letzten Crawl.

Felder

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `SUCCEEDED` | `CANCELLED` | `FAILED`).

  Status des letzten Crawls.

- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Fehlerinformationen über den letzten Crawl, wenn ein Fehler aufgetreten ist.

- `LogGroup` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log group string pattern.

  Die Protokollgruppe für den letzten Crawl.

- `LogStream` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log-stream string pattern.

  Der Protokollstream für den letzten Crawl.

- `MessagePrefix` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Das Präfix für eine Nachricht zu diesem Crawl.

- `StartTime` – Zeitstempel.

  Die Zeit, zu der der Crawl gestartet wurde.

RecrawlPolicy Struktur

Beim Crawling einer Amazon-S3-Datenquelle nach Abschluss des ersten Crawls gibt diese an, ob der gesamte Datensatz erneut gecrawlt werden soll oder nur Ordner gecrawlt werden, die seit der letzten Crawler-Ausführung hinzugefügt wurden. Weitere Informationen finden Sie unter Inkrementelle Crawls in AWS Glue im Entwicklerhandbuch.

Felder

- `RecrawlBehavior` – UTF-8-Zeichenfolge (zulässige Werte: `CRAWL_EVERYTHING` | `CRAWL_NEW_FOLDERS_ONLY` | `CRAWL_EVENT_MODE`).

  Gibt an, ob das Crawling für den gesamten Datensatz erneut ausgeführt werden soll oder nur Ordner gecrawlt werden sollen, die seit der letzten Crawler-Ausführung hinzugefügt wurden.

  Der Wert von `CRAWL_EVERYTHING` gibt an, dass das Crawling des gesamten Dataset erneut ausgeführt wird.

Der Wert von `CRAWL_NEW_FOLDERS_ONLY` gibt an, dass das Crawling nur für Ordner ausgeführt wird, die seit der letzten Crawler-Ausführung hinzugefügt wurden.

Ein Wert von `CRAWL_EVENT_MODE` gibt an, dass nur die durch Amazon S3-Ereignisse identifizierten Änderungen gecrawlt werden.

LineageConfiguration Struktur

Gibt die Konfigurationseinstellungen der Datenherkunft für den Crawler an.

Felder

- `CrawlerLineageSettings` – UTF-8-Zeichenfolge (zulässige Werte: `ENABLE` | `DISABLE`).

  Gibt an, ob die Datenherkunft für den Crawler aktiviert ist. Gültige Werte für sind:

  - ENABLE (AKTIVIEREN): Aktiviert die Datenherkunft für den Crawler
  - DISABLE (DEAKTIVIEREN): Deaktiviert die Datenherkunft für den Crawler

LakeFormationConfiguration Struktur

Gibt die AWS Lake Formation Konfigurationseinstellungen für den Crawler an.

Felder

- `UseLakeFormationCredentials` – Boolesch.

  Gibt an, ob AWS Lake Formation Anmeldeinformationen für den Crawler anstelle der Anmeldeinformationen der IAM-Rolle verwendet werden sollen.
- `AccountId` – UTF-8-Zeichenfolge, nicht mehr als 12 Bytes lang.

  Für kontoübergreifende Crawls erforderlich. Für dieselben Konto-Crawls wie die Zieldaten kann dies auf null belassen werden.

Operationen

- CreateCrawler Aktion (Python: create_crawler)
- DeleteCrawler Aktion (Python: delete_crawler)

- GetCrawler Aktion (Python: get_crawler)
- GetCrawlers Aktion (Python: get_crawlers)
- GetCrawlerMetrics Aktion (Python: get_crawler_metrics)
- UpdateCrawler Aktion (Python: update_crawler)
- StartCrawler Aktion (Python: start_crawler)
- StopCrawler Aktion (Python: stop_crawler)
- BatchGetCrawlers Aktion (Python: batch_get_crawlers)
- ListCrawlers Aktion (Python: list_crawlers)
- ListCrawls Aktion (Python: list_crawls)

CreateCrawler Aktion (Python: create_crawler)

Erstellt einen neuen Crawler mit angegebenen Zielen, Rolle, Konfiguration und optionaler Planung. Mindestens ein Crawl-Ziel muss im Felds`3Targets`, `jdbcTargets` oder `DynamoDBTargets` angegeben werden.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des neuen Crawlers.
- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Die IAM-Rolle oder der Amazon-Ressourcenname (ARN) einer IAM-Rolle, die vom neuen Crawler für den Zugriff auf Kundenressourcen verwendet wird.
- `DatabaseName` – UTF-8-Zeichenfolge.

  Die AWS Glue Datenbank, in die Ergebnisse geschrieben werden, z. B.:. `arn:aws:daylight:us-east-1::database/sometable/*`
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des neuen Crawlers.
- `Targets` – Erforderlich: Ein CrawlerTargets-Objekt.

Eine Liste der Sammlungen von Zielen zum Crawlen.

- `Schedule` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

- `Classifiers` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der benutzerdefinierten Classifier, die der Benutzer registriert hat. Standardmäßig werden alle integrierten Classifier in einen Crawl eingeschlossen. Diese benutzerdefinierten Classifier überschreiben allerdings immer die Standard-Classifier für eine bestimmte Klassifizierung.

- `TablePrefix` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Das Tabellenpräfix, das für erstellte Katalogtabellen verwendet wird.

- `SchemaChangePolicy` – Ein SchemaChangePolicy-Objekt.

  Richtlinie für das Verhalten des Crawlers beim Aktualisieren und Löschen.

- `RecrawlPolicy` – Ein RecrawlPolicy-Objekt.

  Eine Richtlinie, die angibt, ob das Crawling für den gesamten Datensatz erneut ausgeführt werden soll oder nur Ordner gecrawlt werden sollen, die seit der letzten Crawler-Ausführung hinzugefügt wurden.

- `LineageConfiguration` – Ein LineageConfiguration-Objekt.

  Gibt die Konfigurationseinstellungen der Datenherkunft für den Crawler an.

- `LakeFormationConfiguration` – Ein LakeFormationConfiguration-Objekt.

  Gibt die AWS Lake Formation Konfigurationseinstellungen für den Crawler an.

- `Configuration` – UTF-8-Zeichenfolge.

  Crawler-Konfigurationsinformationen. Mit dieser versionierten JSON-Zeichenfolge können Benutzer Verhaltensaspekte eines Crawlers angeben. Weitere Informationen finden Sie unter Festlegen von Crawler-Konfigurationsoptionen.

- `CrawlerSecurityConfiguration` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

Der Name der `SecurityConfiguration` Struktur, die von diesem Crawler verwendet werden soll.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die mit dieser Crawler-Anforderung zu verwendeten Tags. Sie können Tags verwenden, um den Zugriff auf den Crawler einzuschränken. Weitere Informationen zu Tags in AWS Glue finden Sie unter AWS Tags in AWS Glue im Entwicklerhandbuch.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteCrawler Aktion (Python: delete_crawler)

Entfernt einen angegebenen Crawler aus dem AWS Glue Data Catalog, sofern der Crawler-Status nicht lautet. `RUNNING`

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Crawlers, der entfernt werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `CrawlerRunningException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

GetCrawler Aktion (Python: get_crawler)

Ruft Metadaten für einen angegebenen Crawler ab.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Crawlers, für den Metadaten abgerufen werden sollen.

Antwort

- `Crawler` – Ein Crawler-Objekt.

  Die Metadaten für den angegebenen Crawler.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`

GetCrawlers Aktion (Python: get_crawlers)

Ruft Metadaten für alle Crawler ab, die im Kundenkonto definiert sind.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die Anzahl der Crawler, die bei jedem Aufruf zurückgegeben werden sollen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.

Antwort

- `Crawlers` – Ein Array mit Crawler-Objekten.

  Eine Liste der Crawler-Metadaten.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste das Ende der in diesem Kundenkonto definierten Werte nicht erreicht hat.

Fehler

- `OperationTimeoutException`

GetCrawlerMetrics Aktion (Python: get_crawler_metrics)

Ruft Metriken zu angegebenen Crawlern ab.

Anforderung

- `CrawlerNameList` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste mit Namen der Crawler, zu denen Metriken abgerufen werden sollen.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `CrawlerMetricsList` – Ein Array mit CrawlerMetrics-Objekten.

  Eine Liste der Metriken für den angegebenen Crawler.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `OperationTimeoutException`

UpdateCrawler Aktion (Python: update_crawler)

Aktualisiert einen Crawler. Wenn ein Crawler ausgeführt wird, müssen Sie ihn mit `StopCrawler` anhalten, bevor Sie ihn aktualisieren.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des neuen Crawlers.
- `Role` – UTF-8-Zeichenfolge.

  Die IAM-Rolle oder der Amazon-Ressourcenname (ARN) einer IAM-Rolle, die vom neuen Crawler für den Zugriff auf Kundenressourcen verwendet wird.
- `DatabaseName` – UTF-8-Zeichenfolge.

  Die AWS Glue Datenbank, in der Ergebnisse gespeichert werden, z. B.:. `arn:aws:daylight:us-east-1::database/sometable/*`
- `Description` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des neuen Crawlers.
- `Targets` – Ein CrawlerTargets-Objekt.

  Eine Liste der Ziele zum Crawlen.

- `Schedule` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

- `Classifiers` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der benutzerdefinierten Classifier, die der Benutzer registriert hat. Standardmäßig werden alle integrierten Classifier in einen Crawl eingeschlossen. Diese benutzerdefinierten Classifier überschreiben allerdings immer die Standard-Classifier für eine bestimmte Klassifizierung.

- `TablePrefix` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Das Tabellenpräfix, das für erstellte Katalogtabellen verwendet wird.

- `SchemaChangePolicy` – Ein SchemaChangePolicy-Objekt.

  Richtlinie für das Verhalten des Crawlers beim Aktualisieren und Löschen.

- `RecrawlPolicy` – Ein RecrawlPolicy-Objekt.

  Eine Richtlinie, die angibt, ob das Crawling für den gesamten Datensatz erneut ausgeführt werden soll oder nur Ordner gecrawlt werden sollen, die seit der letzten Crawler-Ausführung hinzugefügt wurden.

- `LineageConfiguration` – Ein LineageConfiguration-Objekt.

  Gibt die Konfigurationseinstellungen der Datenherkunft für den Crawler an.

- `LakeFormationConfiguration` – Ein LakeFormationConfiguration-Objekt.

  Gibt die AWS Lake Formation Konfigurationseinstellungen für den Crawler an.

- `Configuration` – UTF-8-Zeichenfolge.

  Crawler-Konfigurationsinformationen. Mit dieser versionierten JSON-Zeichenfolge können Benutzer Verhaltensaspekte eines Crawlers angeben. Weitere Informationen finden Sie unter Festlegen von Crawler-Konfigurationsoptionen.

- `CrawlerSecurityConfiguration` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Der Name der `SecurityConfiguration` Struktur, die von diesem Crawler verwendet werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `VersionMismatchException`
- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StartCrawler Aktion (Python: start_crawler)

Startet einen Crawl mit dem angegebenen Crawler, unabhängig davon, was geplant ist. Wenn der Crawler bereits läuft, wird a zurückgegeben. CrawlerRunningException

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des zu startenden Crawlers.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StopCrawler Aktion (Python: stop_crawler)

Wenn der angegebene Crawler ausgeführt wird, wird der Crawl gestoppt.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des zu stoppenden Crawlers.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `CrawlerNotRunningException`
- `CrawlerStoppingException`
- `OperationTimeoutException`

BatchGetCrawlers Aktion (Python: batch_get_crawlers)

Gibt eine Liste von Ressourcen-Metadaten für eine bestimmte Liste von Crawler-Namen zurück. Nach dem Aufrufen der `ListCrawlers`-Operation können Sie diese Operation aufrufen, um auf die Daten zuzugreifen, für die Ihnen Berechtigungen erteilt wurden. Dieser Vorgang unterstützt alle IAM-Berechtigungen, einschließlich Berechtigungsbedingungen, die Tags verwenden.

Anforderung

- `CrawlerNames` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste von Crawler-Namen, die von der `ListCrawlers`-Operation als Namen zurückgegeben werden können.

Antwort

- `Crawlers` – Ein Array mit Crawler-Objekten.

  Eine Liste der Crawler-Definitionen.

- `CrawlersNotFound` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste mit Namen der Crawler, die nicht gefunden werden konnten.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`

ListCrawlers Aktion (Python: list_crawlers)

Ruft die Namen aller Crawler-Ressourcen in diesem AWS Konto oder der Ressourcen mit dem angegebenen Tag ab. Mit dieser Operation können Sie sehen, welche Ressourcen in Ihrem Konto verfügbar sind, sowie deren Namen.

Diese Operation akzeptiert das optionale `Tags`-Feld, das Sie als Filter für die Antwort verwenden können, so dass markierte Ressourcen als Gruppe abgerufen werden können. Wenn Sie die Tag-Filterung verwenden, werden nur Ressourcen mit dem Tag abgerufen.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Gibt an, das nur diese markierten Ressourcen zurückgegeben werden sollen.

Antwort

- `CrawlerNames` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

Die Namen aller Crawler im Konto oder der Crawler mit den angegebenen Tags.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `OperationTimeoutException`

ListCrawls Aktion (Python: list_crawls)

Gibt alle Crawls eines bestimmten Crawlers zurück. Gibt nur die Crawls zurück, die seit dem Startdatum des Crawler-Verlaufs-Features stattgefunden haben, und behält Crawls nur bis zu 12 Monaten bei. Ältere Crawls werden nicht zurückgegeben.

Sie können diese API verwenden, um:

- Ruft alle Crawls eines bestimmten Crawlers ab.
- Rufen Sie alle Crawls eines bestimmten Crawlers innerhalb einer begrenzten Anzahl ab.
- Rufen Sie alle Crawls eines bestimmten Crawlers in einem bestimmten Zeitraum ab.
- Rufen Sie alle Crawls eines angegebenen Crawlers mit einem bestimmten Status, einer bestimmten Crawl-ID oder einem DPU-Stundenwert ab.

Anforderung

- `CrawlerName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Crawlers, dessen Ausführungen Sie abrufen möchten.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse. Der Standardwert ist 20 und das Maximum ist 100.

- `Filters` – Ein Array mit CrawlsFilter-Objekten.

  Filtert die Crawls nach den Kriterien, die Sie in einer Liste von `CrawlsFilter`-Objekten angeben.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `Crawls` – Ein Array mit CrawlerHistory-Objekten.

  Eine Liste von `CrawlerHistory`-Objekten, die die Ausführung der Crawls repräsentieren, die Ihre Kriterien erfüllen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

API für Spaltenstatistiken

Die API AWS Glue APIs für Spaltenstatistiken beschreibt die Rückgabe von Statistiken zu Spalten in einer Tabelle.

Datentypen

- ColumnStatisticsTaskRun Struktur
- ColumnStatisticsTaskSettings Struktur
- ExecutionAttempt Struktur

ColumnStatisticsTaskRun Struktur

Das Objekt, das die Details der Ausführung von Spaltenstatistiken anzeigt.

Felder

- `CustomerId` – UTF-8-Zeichenfolge, nicht mehr als 12 Bytes lang.

  Die AWS Konto-ID.

- `ColumnStatisticsTaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Bezeichner für die jeweilige Ausführung der Spaltenstatistik-Aufgabe.

- `DatabaseName` – UTF-8-Zeichenfolge.

  Die Datenbank, in der die Tabelle gespeichert ist.

- `TableName` – UTF-8-Zeichenfolge.

  Der Name der Tabelle, für die Spaltenstatistiken generiert werden.

- `ColumnNameList` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Spaltennamen. Wird keiner angegeben, werden standardmäßig alle Spaltennamen für die Tabelle verwendet.

- `CatalogID` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Wenn keine angegeben wird, wird standardmäßig die AWS Konto-ID verwendet.

- `Role` – UTF-8-Zeichenfolge.

  Die IAM-Rolle, die der Service zur Generierung von Statistiken übernimmt.

- `SampleSize` – Zahl (Double), nicht mehr als 100.

  Der Prozentsatz der Zeilen, die zur Generierung von Statistiken verwendet wurden. Wird nichts angegeben, wird die gesamte Tabelle verwendet, um Statistiken zu generieren.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Name der Sicherheitskonfiguration, die zum Verschlüsseln von CloudWatch Protokollen für die Ausführung der Aufgabe mit Spaltenstatistiken verwendet wird.

- `NumberOfWorkers` – Zahl (ganze Zahl), mindestens 1.

Die Anzahl der Worker, die zur Generierung von Spaltenstatistiken verwendet werden. Der Auftrag ist für die automatische Skalierung auf bis zu 25 Instances vorkonfiguriert.

- `WorkerType` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Art der Worker, die für die Generierung von Statistiken verwendet werden. Der Standardwert ist `g.1x`.

- `ComputationType` – UTF-8-Zeichenfolge (zulässige Werte: `FULL` | `INCREMENTAL`).

  Die Art der Berechnung der Spaltenstatistiken.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `STARTING` | `RUNNING` | `SUCCEEDED` | `FAILED` | `STOPPED`).

  Der Status der ausgeführten Aufgabe.

- `CreationTime` – Zeitstempel.

  Die Zeit, zu der diese Aufgabe erstellt wurde.

- `LastUpdated` – Zeitstempel.

  Der Zeitpunkt, an diese Aufgabe zuletzt geändert wurde.

- `StartTime` – Zeitstempel.

  Die Anfangszeit der Aufgabe.

- `EndTime` – Zeitstempel.

  Die Endzeit der Aufgabe.

- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Fehlermeldung für den Auftrag.

- `DPUSeconds` – Zahl (Double), nicht mehr als Keine.

  Die berechnete DPU-Auslastung in Sekunden für alle automatisch skalierten Worker.

ColumnStatisticsTaskSettings Struktur

Die Einstellungen für eine Aufgabe mit Spaltenstatistiken.

Felder

- `DatabaseName` – UTF-8-Zeichenfolge.

  Der Name der Datenbank, in der sich die Tabelle befindet.

- `TableName` – UTF-8-Zeichenfolge.

  Der Name der Tabelle, für die Spaltenstatistiken generiert werden sollen.

- `Schedule` – Ein Plan-Objekt.

  Ein Zeitplan für die Ausführung der Spaltenstatistiken, angegeben in der CRON-Syntax.

- `ColumnNameList` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Spaltennamen, für die Statistiken ausgeführt werden sollen.

- `CatalogID` – Katalog-ID-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist.

- `Role` – UTF-8-Zeichenfolge.

  Die Rolle, die für die Ausführung der Spaltenstatistiken verwendet wird.

- `SampleSize` – Zahl (Double), nicht mehr als 100.

  Der Prozentsatz der zu sammelnden Daten.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht mehr als 128 Bytes lang.

  Name der Sicherheitskonfiguration, die zum Verschlüsseln von CloudWatch Protokollen verwendet wird.

- `ScheduleType` – UTF-8-Zeichenfolge (zulässige Werte: `CRON` | `AUTO`).

  Der Typ des Zeitplans für eine Aufgabe mit Spaltenstatistiken. Mögliche Werte können `CRON` oder sein`AUTO`.

- `SettingSource` – UTF-8-Zeichenfolge (zulässige Werte: `CATALOG` | `TABLE`).

  Die Quelle für die Einstellung der Aufgabe „Spaltenstatistik“. Mögliche Werte können `CATALOG` oder sein`TABLE`.

- `LastExecutionAttempt` – Ein ExecutionAttempt-Objekt.

Die letzte Ausführung der Aufgabe `ExecutionAttempt` für die Spaltenstatistik.

ExecutionAttempt Struktur

Ein Ausführungsversuch für die Ausführung einer Aufgabe mit Spaltenstatistiken.

Felder

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `FAILED` | `STARTED`).

  Der Status der letzten Ausführung der Aufgabe für Spaltenstatistiken.
- `ColumnStatisticsTaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Aufgabenlauf-ID für die letzte Ausführung der Aufgabe mit Spaltenstatistiken.
- `ExecutionTimestamp` – Zeitstempel.

  Ein Zeitstempel, zu dem die letzte Ausführung der Aufgabe für Spaltenstatistiken stattgefunden hat.
- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Fehlermeldung im Zusammenhang mit der letzten Ausführung der Aufgabe für Spaltenstatistiken.

Operationen

- StartColumnStatisticsTaskRun Aktion (Python: start_column_statistics_task_run)
- GetColumnStatisticsTaskRun Aktion (Python: get_column_statistics_task_run)
- GetColumnStatisticsTaskRuns Aktion (Python: get_column_statistics_task_runs)
- ListColumnStatisticsTaskRuns Aktion (Python: list_column_statistics_task_runs)
- StopColumnStatisticsTaskRun Aktion (Python: stop_column_statistics_task_run)
- CreateColumnStatisticsTaskSettings Aktion (Python: create_column_statistics_task_settings)
- UpdateColumnStatisticsTaskSettings Aktion (Python: update_column_statistics_task_settings)
- GetColumnStatisticsTaskSettings Aktion (Python: get_column_statistics_task_settings)
- DeleteColumnStatisticsTaskSettings Aktion (Python: delete_column_statistics_task_settings)

- StartColumnStatisticsTaskRunSchedule Aktion (Python: start_column_statistics_task_run_schedule)
- StopColumnStatisticsTaskRunSchedule Aktion (Python: stop_column_statistics_task_run_schedule)

StartColumnStatisticsTaskRun Aktion (Python: start_column_statistics_task_run)

Startet eine Aufgabenausführung für Spaltenstatistiken für eine angegebene Tabelle und ihre Spalten.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle zum Generieren von Statistiken.
- `ColumnNameList` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Spaltennamen, um Statistiken zu generieren. Wird keiner angegeben, werden standardmäßig alle Spaltennamen für die Tabelle verwendet.
- `Role` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die IAM-Rolle, die der Service zur Generierung von Statistiken übernimmt.
- `SampleSize` – Zahl (Double), nicht mehr als 100.

  Der Prozentsatz der Zeilen, die zur Generierung von Statistiken verwendet wurden. Wird nichts angegeben, wird die gesamte Tabelle verwendet, um Statistiken zu generieren.
- `CatalogID` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem sich die Tabelle befindet. Ist keine ID bereitgestellt, wird standardmäßig die AWS -Konto-ID verwendet.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name der Sicherheitskonfiguration, die zum Verschlüsseln von Protokollen für die Ausführung der Aufgabe mit Spaltenstatistiken verwendet wird. CloudWatch

Antwort

- `ColumnStatisticsTaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Bezeichner für die Ausführung der Spaltenstatistik-Aufgabe.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `ColumnStatisticsTaskRunningException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InvalidInputException`

GetColumnStatisticsTaskRun Aktion (Python: get_column_statistics_task_run)

Ruft die zugehörigen Metadaten/Informationen für eine Aufgabenausführung ab, wenn eine Aufgabenausführungs-ID angegeben ist.

Anforderung

- `ColumnStatisticsTaskRunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Bezeichner für die jeweilige Ausführung der Spaltenstatistik-Aufgabe.

Antwort

- `ColumnStatisticsTaskRun` – Ein ColumnStatisticsTaskRun-Objekt.

Ein `ColumnStatisticsTaskRun`-Objekt, das die Details der Ausführung mit Spaltenstatistiken darstellt.

Fehler

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

GetColumnStatisticsTaskRuns Aktion (Python: get_column_statistics_task_runs)

Ruft Informationen zu allen Ausführungen in Verbindung mit der angegebenen Tabelle ab.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der Antwort.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `ColumnStatisticsTaskRuns` – Ein Array mit ColumnStatisticsTaskRun-Objekten.

  Eine Liste der Spaltenstatistik-Aufgabenausführungen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungstoken, falls noch nicht alle Aufgabenausführungen zurückgegeben wurden.

Fehler

- `OperationTimeoutException`

ListColumnStatisticsTaskRuns Aktion (Python: list_column_statistics_task_runs)

Listet alle Aufgabenausführungen für ein bestimmtes Konto auf.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der Antwort.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `ColumnStatisticsTaskRunIds` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 100 Zeichenfolgen.

  Eine Liste der ausgeführten Aufgaben mit Spaltenstatistiken. IDs
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungstoken, falls noch nicht alle IDs ausgeführten Aufgaben zurückgegeben wurden.

Fehler

- `OperationTimeoutException`

StopColumnStatisticsTaskRun Aktion (Python: stop_column_statistics_task_run)

Stoppt die Ausführung einer Aufgabe für die angegebene Tabelle.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name der Datenbank, in der sich die Tabelle befindet.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `ColumnStatisticsTaskNotRunningException`
- `ColumnStatisticsTaskStoppingException`
- `OperationTimeoutException`

CreateColumnStatisticsTaskSettings Aktion (Python: create_column_statistics_task_settings)

Erstellt Einstellungen für eine Aufgabe mit Spaltenstatistiken.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die Spaltenstatistiken generiert werden sollen.
- `Role` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Rolle, die für die Ausführung der Spaltenstatistiken verwendet wird.
- `Schedule` – UTF-8-Zeichenfolge.

Ein Zeitplan für die Ausführung der Spaltenstatistiken, angegeben in der CRON-Syntax.

- `ColumnNameList` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Spaltennamen, für die Statistiken ausgeführt werden sollen.

- `SampleSize` – Zahl (Double), nicht mehr als 100.

  Der Prozentsatz der zu sammelnden Daten.

- `CatalogID` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name der Sicherheitskonfiguration, die zum Verschlüsseln von CloudWatch Protokollen verwendet wird.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Eine Übersicht von Stichwörtern.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

- `ResourceNumberLimitExceededException`
- `ColumnStatisticsTaskRunningException`

UpdateColumnStatisticsTaskSettings Aktion (Python: update_column_statistics_task_settings)

Aktualisiert die Einstellungen für eine Aufgabe mit Spaltenstatistiken.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die Spaltenstatistiken generiert werden sollen.
- `Role` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Rolle, die für die Ausführung der Spaltenstatistiken verwendet wird.
- `Schedule` – UTF-8-Zeichenfolge.

  Ein Zeitplan für die Ausführung der Spaltenstatistiken, angegeben in der CRON-Syntax.
- `ColumnNameList` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Spaltennamen, für die Statistiken ausgeführt werden sollen.
- `SampleSize` – Zahl (Double), nicht mehr als 100.

  Der Prozentsatz der zu sammelnden Daten.
- `CatalogID` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Data Catalog, in dem die Datenbank gespeichert ist.
- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Name der Sicherheitskonfiguration, die zum Verschlüsseln von CloudWatch Protokollen verwendet wird.

Antwort

- Keine Antwortparameter.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`
- `VersionMismatchException`
- `OperationTimeoutException`

GetColumnStatisticsTaskSettings Aktion (Python: get_column_statistics_task_settings)

Ruft Einstellungen für eine Aufgabe mit Spaltenstatistiken ab.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die Spaltenstatistiken abgerufen werden sollen.

Antwort

- `ColumnStatisticsTaskSettings` – Ein ColumnStatisticsTaskSettings-Objekt.

  Ein `ColumnStatisticsTaskSettings` Objekt, das die Einstellungen für die Aufgabe zur Spaltenstatistik darstellt.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

DeleteColumnStatisticsTaskSettings Aktion (Python: delete_column_statistics_task_settings)

Löscht Einstellungen für eine Aufgabe mit Spaltenstatistiken.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die Spaltenstatistiken gelöscht werden sollen.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

StartColumnStatisticsTaskRunSchedule Aktion (Python: start_column_statistics_task_run_schedule)

Startet einen Zeitplan für die Ausführung einer Aufgabe mit Spaltenstatistiken.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Tabelle, für die ein Zeitplan für die Ausführung einer Aufgabe für Spaltenstatistiken gestartet werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

StopColumnStatisticsTaskRunSchedule Aktion (Python: stop_column_statistics_task_run_schedule)

Stoppt den Zeitplan für die Ausführung einer Aufgabe mit Spaltenstatistiken.

Anforderung

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der sich die Tabelle befindet.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Tabelle, für die der Zeitplan für die Ausführung einer Aufgabe zur Spaltenstatistik beendet werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

Ausnahmen

- ColumnStatisticsTaskRunningException Struktur
- ColumnStatisticsTaskNotRunningException Struktur
- ColumnStatisticsTaskStoppingException Struktur
- ColumnStatisticsTaskAutoConcurrencyLimitException Struktur
- InvalidCatalogSettingException Struktur

ColumnStatisticsTaskRunningException Struktur

Eine Ausnahme, die ausgelöst wird, wenn Sie versuchen, einen anderen Auftrag zu starten, während Sie einen Auftrag zur Generierung von Spaltenstatistiken ausführen.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ColumnStatisticsTaskNotRunningException Struktur

Eine Ausnahme, die ausgelöst wird, wenn Sie versuchen, eine Aufgabenausführung zu beenden, obwohl keine Aufgabe ausgeführt wird.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ColumnStatisticsTaskStoppingException Struktur

Eine Ausnahme, die ausgelöst wird, wenn Sie versuchen, eine Aufgabenausführung zu beenden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ColumnStatisticsTaskAutoConcurrencyLimitException Struktur

Eine Ausnahme, die ausgelöst wird, wenn Sie das Limit für gleichzeitige auto Statistikjobs bereits erreicht haben.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InvalidCatalogSettingException Struktur

Eine Ausnahme, die ausgelöst wird, wenn ein Problem mit den Katalogeinstellungen auftritt.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

Crawler-Scheduler-API

Die Crawler-Scheduler-API beschreibt AWS Glue Crawler-Datentypen sowie die API zum Erstellen, Löschen, Aktualisieren und Auflisten von Crawlern.

Datentypen

- Planstruktur

Planstruktur

Ein Planungsobjekt, das eine `cron`-Anweisung zum Planen eines Ereignisses verwendet.

Felder

- `ScheduleExpression` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.
- `State` – UTF-8-Zeichenfolge (zulässige Werte: `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  Der Status des Zeitplans.

Operationen

- UpdateCrawlerSchedule Aktion (Python: update_crawler_schedule)
- StartCrawlerSchedule Aktion (Python: start_crawler_schedule)
- StopCrawlerSchedule Aktion (Python: stop_crawler_schedule)

UpdateCrawlerSchedule Aktion (Python: update_crawler_schedule)

Aktualisiert den Zeitplan eines Crawlers über einen `cron`-Ausdruck.

Anforderung

- `CrawlerName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Name des Crawlers, dessen Zeitplan aktualisiert werden soll.

- `Schedule` – UTF-8-Zeichenfolge.

  Der aktualisierte `cron`-Ausdruck, der verwendet wird, um den Zeitplan anzugeben (siehe Zeitpläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `VersionMismatchException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

StartCrawlerSchedule Aktion (Python: start_crawler_schedule)

Ändert den Planungsstatus des angegebenen Crawlers zu `SCHEDULED`, sofern der Crawler nicht bereits ausgeführt wird oder der Planungsstatus nicht bereits `SCHEDULED` lautet.

Anforderung

- `CrawlerName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Crawlers, der geplant werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `SchedulerRunningException`
- `SchedulerTransitioningException`
- `NoScheduleException`
- `OperationTimeoutException`

StopCrawlerSchedule Aktion (Python: stop_crawler_schedule)

Legt den Planungsstatus des angegebenen Crawlers auf `NOT_SCHEDULED` fest, stoppt den Crawler jedoch nicht, wenn er bereits ausgeführt wird.

Anforderung

- `CrawlerName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Crawlers, dessen Planungsstatus festgelegt werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `SchedulerNotRunningException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

Automatisches Generieren einer API für ETL-Skripts

Die API für die Generierung von ETL-Skripts beschreibt die Datentypen und die API zur Generierung von ETL-Skripts in AWS Glue.

Datentypen

- CodeGenNode Struktur
- CodeGenNodeArg Struktur
- CodeGenEdge Struktur
- Speicherortstruktur
- CatalogEntry Struktur
- MappingEntry Struktur

CodeGenNode Struktur

Repräsentiert einen Knoten in einem azyklisch gerichteten Diagramm (DAG)

Felder

- `Id` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Identifier string pattern.

  Eine Knotenkennung, die innerhalb des Knotendiagramms einzigartig ist.
- `NodeType` – Erforderlich: UTF-8-Zeichenfolge.

  Der Typ des Knotens, der hier vorliegt.
- `Args` – Erforderlich: Ein Array mit CodeGenNodeArg-Objekten, nicht mehr als 50 Strukturen.

  Eigenschaften des Knotens in Form von Name-Wert-Paaren.
- `LineNumber` – Zahl (Ganzzahl).

  Die Zeilennummer des Knotens.

CodeGenNodeArg Struktur

Ein Argument oder eine Eigenschaft eines Knotens.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge.

Der Name des Arguments oder der Eigenschaft.

- `Value` – Erforderlich: UTF-8-Zeichenfolge.

  Der Wert des Arguments oder der Eigenschaft.

- `Param` – Boolesch.

  True, wenn der Wert als Parameter verwendet wird.

CodeGenEdge Struktur

Repräsentiert einen Richtungs-Edge in einem azyklisch gerichteten Diagramm (DAG).

Felder

- `Source` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Identifier string pattern.

  Die ID des Knotens, an dem der Edge beginnt.

- `Target` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Identifier string pattern.

  Die ID des Knotens, an dem der Edge endet.

- `TargetParameter` – UTF-8-Zeichenfolge.

  Das Ziel des Edge.

Speicherortstruktur

Der Speicherort von Ressourcen.

Felder

- `Jdbc` – Ein Array mit CodeGenNodeArg-Objekten, nicht mehr als 50 Strukturen.

  Ein JDBC-Speicherort.

- `S3` – Ein Array mit CodeGenNodeArg-Objekten, nicht mehr als 50 Strukturen.

  Amazon Simple Storage Service (Amazon S3)-Speicherort.

- `DynamoDB` – Ein Array mit CodeGenNodeArg-Objekten, nicht mehr als 50 Strukturen.

  Ein Amazon DynamoDB-Tabellenspeicherort.

CatalogEntry Struktur

Gibt eine Tabellendefinition im AWS Glue Data Catalog an.

Felder

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Datenbank, in der die Tabellenmetadaten gespeichert sind.
- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der betreffenden Tabelle.

MappingEntry Struktur

Definiert ein Mapping.

Felder

- `SourceTable` – UTF-8-Zeichenfolge.

  Der Name der Quelltabelle.
- `SourcePath` – UTF-8-Zeichenfolge.

  Der -Quellpfad
- `SourceType` – UTF-8-Zeichenfolge.

  Der Quelltyp.
- `TargetTable` – UTF-8-Zeichenfolge.

  Die Zieltabelle.
- `TargetPath` – UTF-8-Zeichenfolge.

Der Zielpfad.

- `TargetType` – UTF-8-Zeichenfolge.

  Der Zieltyp.

Operationen

- CreateScript Aktion (Python: create_script)
- GetDataflowGraph Aktion (Python: get_dataflow_graph)
- GetMapping Aktion (Python: get_mapping)
- GetPlan Aktion (Python: get_plan)

CreateScript Aktion (Python: create_script)

Wandelt ein azyklisch gerichtetes Diagramm (DAG) in Code um.

Anforderung

- `DagNodes` – Ein Array mit CodeGenNode-Objekten.

  Eine Liste der Knoten im DAG.

- `DagEdges` – Ein Array mit CodeGenEdge-Objekten.

  Eine Liste der Edges im DAG.

- `Language` – UTF-8-Zeichenfolge (zulässige Werte: `PYTHON` | `SCALA`).

  Die Programmiersprache des resultierenden Codes aus dem DAG.

Antwort

- `PythonScript` – UTF-8-Zeichenfolge.

  Das aus dem DAG generierte Python-Skript.

- `ScalaCode` – UTF-8-Zeichenfolge.

  Der aus dem DAG generierte Scala-Code.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetDataflowGraph Aktion (Python: get_dataflow_graph)

Wandelt ein Python-Skript in ein azyklisch gerichtetes Diagramm (DAG) um.

Anforderung

- `PythonScript` – UTF-8-Zeichenfolge.

  Das umzuwandelnde Python-Skript.

Antwort

- `DagNodes` – Ein Array mit CodeGenNode-Objekten.

  Eine Liste der Knoten im resultierenden DAG.
- `DagEdges` – Ein Array mit CodeGenEdge-Objekten.

  Eine Liste der Edges im resultierenden DAG.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetMapping Aktion (Python: get_mapping)

Erstellt Mappings.

Anforderung

- `Source` – Erforderlich: Ein CatalogEntry-Objekt.

Gibt die Quelltabelle an.

- `Sinks` – Ein Array mit CatalogEntry-Objekten.

  Eine Liste der Zieltabellen.

- `Location` – Ein Ort-Objekt.

  Parameter für das Mapping.

Antwort

- `Mapping` – Erforderlich: Ein Array mit MappingEntry-Objekten.

  Eine Liste der Mappings zu den angegebenen Zielen.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

GetPlan Aktion (Python: get_plan)

Ruft Code zum Ausführen eines angegebenen Mappings ab.

Anforderung

- `Mapping` – Erforderlich: Ein Array mit MappingEntry-Objekten.

  Die Liste der Mappings aus einer Quelltabelle zu Zieltabellen.

- `Source` – Erforderlich: Ein CatalogEntry-Objekt.

  Die Quelltabelle.

- `Sinks` – Ein Array mit CatalogEntry-Objekten.

  Die Zieltabellen.

- `Location` – Ein Ort-Objekt.

  Die Parameter für das Mapping.

- `Language` – UTF-8-Zeichenfolge (zulässige Werte: `PYTHON` | `SCALA`).

  Die Programmiersprache des Codes zum Ausführen des Mappings.

- `AdditionalPlanOptionsMap` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Map, die zusätzliche optionale Schlüssel-Wert-Parameter enthält.

  Derzeit werden diese Schlüssel-Wert-Paare unterstützt:

  - `inferSchema`— Gibt an, ob das von einem Job generierte Standardskript auf true oder false gesetzt werden `inferSchema` soll. AWS Glue Legen Sie beispielsweise `inferSchema` auf „true“ fest, um das folgende Schlüssel-Wert-Paar zu übergeben:

    ```
    --additional-plan-options-map '{"inferSchema":"true"}'
    ```

Antwort

- `PythonScript` – UTF-8-Zeichenfolge.

  Ein Python-Skript zum Ausführen des Mappings.

- `ScalaCode` – UTF-8-Zeichenfolge.

  Der Scala-Code zum Ausführen des Mappings.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

Visual Job API

Die Visual-Job-API ermöglicht es Ihnen, Datenintegrationsjobs mithilfe der AWS Glue API aus einem JSON-Objekt zu erstellen, das eine visuelle Konfiguration eines AWS Glue Jobs darstellt.

Eine Liste von `CodeGenConfigurationNodes` wird einer API zum Erstellen oder Aktualisieren von Jobs zur Verfügung gestellt, um eine DAG in AWS Glue Studio für den erstellten Job zu registrieren und den zugehörigen Code zu generieren.

Datentypen

- CodeGenConfigurationNode Struktur
- JDBCConnectorStruktur der Optionen
- StreamingDataPreviewOptions Struktur
- AthenaConnectorSource Struktur
- JDBCConnectorStruktur der Quelle
- SparkConnectorSource Struktur
- CatalogSource Struktur
- Meine SQLCatalog Quellstruktur
- SQLCatalogPostgre-Quellstruktur
- Oracle-Quellstruktur SQLCatalog
- SQLServerCatalogSource Microsoft-Struktur
- CatalogKinesisSource Struktur
- DirectKinesisSource Struktur
- KinesisStreamingSourceOptions Struktur
- CatalogKafkaSource Struktur
- DirectKafkaSource Struktur
- KafkaStreamingSourceOptions Struktur
- RedshiftSource Struktur
- AmazonRedshiftSource Struktur
- AmazonRedshiftNodeData Struktur

- AmazonRedshiftAdvancedOption Struktur
- Optionsstruktur
- CatalogSource S3-Struktur
- SourceAdditionalOptions S3-Struktur
- CsvSource S3-Struktur
- Direkte JDBCSource Struktur
- DirectSourceAdditionalOptions S3-Struktur
- JsonSource S3-Struktur
- ParquetSource S3-Struktur
- DeltaSource S3-Struktur
- CatalogDeltaSource S3-Struktur
- CatalogDeltaSource Struktur
- HudiSource S3-Struktur
- S3-Struktur CatalogHudiSource
- S3-Struktur ExcelSource
- CatalogHudiSource Struktur
- Dynamo-Quellstruktur DBCatalog
- RelationalCatalogSource Struktur
- JDBCConnectorZielstruktur
- SparkConnectorTarget Struktur
- BasicCatalogTarget Struktur
- Meine SQLCatalog Zielstruktur
- SQLCatalogPostgre-Target-Struktur
- Struktur von Oracle SQLCatalog Target
- SQLServerCatalogTarget Microsoft-Struktur
- RedshiftTarget Struktur
- AmazonRedshiftTarget Struktur
- UpsertRedshiftTargetOptions Struktur
- CatalogTarget S3-Struktur

- GlueParquetTarget S3-Struktur
- CatalogSchemaChangePolicy Struktur
- DirectTarget S3-Struktur
- S3-Struktur HudiCatalogTarget
- S3-Struktur HudiDirectTarget
- S3-Struktur DeltaCatalogTarget
- DeltaDirectTarget S3-Struktur
- S3-Struktur HyperDirectTarget
- IcebergDirectTarget S3-Struktur
- DirectSchemaChangePolicy Struktur
- ApplyMapping Struktur
- Mapping-Struktur
- SelectFields Struktur
- DropFields Struktur
- RenameField Struktur
- Spigot-Struktur
- Join-Struktur
- JoinColumn Struktur
- SplitFields Struktur
- SelectFromCollection Struktur
- FillMissingValues Struktur
- Filter-Struktur
- FilterExpression Struktur
- FilterValue Struktur
- CustomCode Struktur
- SparkSQL-Struktur
- SqlAlias Struktur
- DropNullFields Struktur
- NullCheckBoxList Struktur

- NullValueField Struktur
- Datatype-Struktur
- Merge-Struktur
- Union-Struktur
- PIIDetection Struktur
- Aggregierte Struktur
- DropDuplicates Struktur
- GovernedCatalogTarget Struktur
- GovernedCatalogSource Struktur
- AggregateOperation Struktur
- GlueSchema Struktur
- GlueStudioSchemaColumn Struktur
- GlueStudioColumn Struktur
- DynamicTransform Struktur
- TransformConfigParameter Struktur
- EvaluateDataQuality Struktur
- DQResultsPublishingOptions Struktur
- DQStopJobOnFailureOptions Struktur
- EvaluateDataQualityMultiFrame Struktur
- Struktur des Rezepts
- RecipeReference Struktur
- SnowflakeNodeData Struktur
- SnowflakeSource Struktur
- SnowflakeTarget Struktur
- ConnectorDataSource Struktur
- ConnectorDataTarget Struktur
- RecipeStep Struktur
- RecipeAction Struktur
- ConditionExpression Struktur

CodeGenConfigurationNode Struktur

`CodeGenConfigurationNode` zählt alle gültigen Knotentypen auf. Nur eine ihrer Membervariablen kann ausgefüllt werden.

Felder

- `AthenaConnectorSource` – Ein AthenaConnectorSource-Objekt.

  Gibt einen Konnektor zu einer Amazon Athena-Datenquelle an.
- `JDBCConnectorSource` – Ein JDBCConnectorQuelle-Objekt.

  Gibt einen Konnektor zu einer JDBC-Datenquelle an.
- `SparkConnectorSource` – Ein SparkConnectorSource-Objekt.

  Gibt einen Konnektor zu einer Apache-Spark-Datenquelle an.
- `CatalogSource` – Ein CatalogSource-Objekt.

  Gibt einen Datenspeicher im AWS Glue Datenkatalog an.
- `RedshiftSource` – Ein RedshiftSource-Objekt.

  Gibt einen Amazon Redshift-Datenspeicher an.
- `S3CatalogSource` – Ein S3 CatalogSource-Objekt.

  Gibt einen Amazon S3 S3-Datenspeicher im AWS Glue Datenkatalog an.
- `S3CsvSource` – Ein S3 CsvSource-Objekt.

  Gibt einen CSV-Datenspeicher an, der in Amazon S3 gespeichert ist.
- `S3ExcelSource` – Ein S3 ExcelSource-Objekt.

  Definiert Konfigurationsparameter für das Lesen von Excel-Dateien aus Amazon S3.
- `S3JsonSource` – Ein S3 JsonSource-Objekt.

  Gibt einen JSON-Datenspeicher an, der in Amazon S3 gespeichert ist.
- `S3ParquetSource` – Ein S3 ParquetSource-Objekt.

  Gibt einen Apache-Parquet-Datenspeicher an, der in Amazon S3 gespeichert ist.
- `RelationalCatalogSource` – Ein RelationalCatalogSource-Objekt.

Gibt einen relationalen Katalogdatenspeicher im AWS Glue Datenkatalog an.

- `DynamoDBCatalogSource` – Ein Dynamo-Quelle DBCatalog-Objekt.

  Gibt einen DynamoDBC-Katalog-Datenspeicher im Datenkatalog an AWS Glue .

- `JDBCConnectorTarget` – Ein JDBCConnectorZiel-Objekt.

  Gibt ein Datenziel an, das in den Säulenspeicher von Apache Parquet in Amazon S3 schreibt.

- `SparkConnectorTarget` – Ein SparkConnectorTarget-Objekt.

  Gibt ein Ziel an, das einen Apache-Spark-Konnektor verwendet.

- `CatalogTarget` – Ein BasicCatalogTarget-Objekt.

  Gibt ein Ziel an, das eine AWS Glue Datenkatalogtabelle verwendet.

- `RedshiftTarget` – Ein RedshiftTarget-Objekt.

  Gibt ein Ziel an, das Amazon Redshift verwendet.

- `S3CatalogTarget` – Ein S3 CatalogTarget-Objekt.

  Gibt ein Datenziel an, das mithilfe des AWS Glue Datenkatalogs in Amazon S3 schreibt.

- `S3GlueParquetTarget` – Ein S3 GlueParquetTarget-Objekt.

  Gibt ein Datenziel an, das in den Säulenspeicher von Apache Parquet in Amazon S3 schreibt.

- `S3HyperDirectTarget` – Ein S3 HyperDirectTarget-Objekt.

  Definiert Konfigurationsparameter für das Schreiben von Daten in Amazon S3 mithilfe von HyperDirect Optimierung.

- `S3DirectTarget` – Ein S3 DirectTarget-Objekt.

  Gibt ein Datenziel an, das an Amazon S3 schreibt.

- `S3IcebergDirectTarget` – Ein S3 IcebergDirectTarget-Objekt.

  Definiert Konfigurationsparameter für das Schreiben von Daten in Amazon S3 als Apache Iceberg-Tabelle.

- `ApplyMapping` – Ein ApplyMapping-Objekt.

  Gibt eine Transformation an, die Dateneigenschaftsschlüssel in der Datenquelle den Dateneigenschaftsschlüsseln im Datenziel zuordnet. Sie können Schlüssel umbenennen, die

Datentypen für Schlüssel ändern und die Schlüssel auswählen, die aus dem Datensatz gelöscht werden sollen.

- `SelectFields` – Ein SelectFields-Objekt.

  Gibt eine Transformation an, die die Dateneigenschaftsschlüssel auswählt, die Sie beibehalten möchten.

- `DropFields` – Ein DropFields-Objekt.

  Gibt eine Transformation an, die die Dateneigenschaftsschlüssel auswählt, die Sie löschen möchten.

- `RenameField` – Ein RenameField-Objekt.

  Gibt eine Transformation an, die einen einzelnen Dateneigenschaftsschlüssel umbenennt.

- `Spigot` – Ein Spigot-Objekt.

  Gibt eine Transformation an, die Beispiele der Daten in einen Amazon S3-Bucket schreibt.

- `Join` – Ein Join-Objekt.

  Gibt eine Transformation an, die zwei Datensätze mit einer Vergleichsphrase für die angegebenen Dateneigenschaftsschlüssel zu einem Datensatz zusammenführt. Sie können innere, äußere, linke, rechte, linke Hälfte und linke Anti-Joins verwenden.

- `SplitFields` – Ein SplitFields-Objekt.

  Gibt eine Transformation an, die Dateneigenschaftsschlüssel in zwei te`DynamicFrames` aufteilt. Die Ausgabe ist eine Sammlung von `DynamicFrames`: Einer mit ausgewählten Dateneigenschaftsschlüsseln und einer mit den übrigen Dateneigenschaftsschlüsseln.

- `SelectFromCollection` – Ein SelectFromCollection-Objekt.

  Gibt eine Transformation an, die einen `DynamicFrame` aus einer Reihe von `DynamicFrames` auswählt. Die Ausgabe ist das ausgewählte `DynamicFrame`.

- `FillMissingValues` – Ein FillMissingValues-Objekt.

  Gibt eine Transformation an, bei der Akten im Datensatz gesucht werden können, die fehlende Werte aufweisen, und die ein neues Feld mit einem durch Imputation bestimmten Wert hinzufügt. Mit dem Eingabedatensatz wird das Modell zum Machine Learning (ML) trainiert, das den fehlenden Wert bestimmt.

- `Filter` – Ein Filter-Objekt.

Gibt eine Transformation an, die einen Datensatz basierend auf einer Filterbedingung in zwei Teile teilt.

- `CustomCode` – Ein CustomCode-Objekt.

  Gibt eine Transformation an, die benutzerdefinierten Code verwendet, den Sie zur Durchführung der Datentransformation bereitstellen. Die Ausgabe ist eine Sammlung von. DynamicFrames

- `SparkSQL` – Ein SparkSQL-Objekt.

  Gibt eine Transformation an, bei der Sie eine SQL-Abfrage mit der Spark SQL-Syntax eingeben, um die Daten zu transformieren. Die Ausgabe ist ein einzelner `DynamicFrame`.

- `DirectKinesisSource` – Ein DirectKinesisSource-Objekt.

  Gibt eine direkte Amazon Kinesis-Datenquelle an.

- `DirectKafkaSource` – Ein DirectKafkaSource-Objekt.

  Gibt einen Apache-Kafka-Datenspeicher an.

- `CatalogKinesisSource` – Ein CatalogKinesisSource-Objekt.

  Gibt eine Kinesis-Datenquelle im AWS Glue Datenkatalog an.

- `CatalogKafkaSource` – Ein CatalogKafkaSource-Objekt.

  Gibt einen Apache-Kafka-Datenspeicher im Data Catalog an.

- `DropNullFields` – Ein DropNullFields-Objekt.

  Gibt eine Transformation an, die Spalten aus dem Datensatz entfernt, wenn alle Werte in der Spalte „null“ sind. Standardmäßig erkennt AWS Glue Studio Null-Objekte, aber einige Werte wie leere Zeichenketten, Zeichenketten, die „Null“ sind, ganze Zahlen vom Typ -1 oder andere Platzhalter wie Nullen, werden nicht automatisch als Nullen erkannt.

- `Merge` – Ein Mischen von-Objekt.

  Gibt eine Transformation an, die ein `DynamicFrame` mit einem Staging-`DynamicFrame` basierend auf den angegebenen Primärschlüsseln zusammenführt, um Datensätze zu identifizieren. Doppelte Datensätze (Datensätze mit denselben Primärschlüsseln) werden nicht dedupliziert.

- `Union` – Ein Union-Objekt.

  Gibt eine Transformation an, die die Zeilen aus zwei oder mehr Datensätzen zu einem einzigen Ergebnis kombiniert.

- `PIIDetection` – Ein PIIDetection-Objekt.

  Gibt eine Transformation an, die PII-Daten identifiziert, entfernt oder maskiert.

- `Aggregate` – Ein Aggregate-Objekt.

  Gibt eine Transformation an, die Zeilen nach ausgewählten Feldern gruppiert und den aggregierten Wert nach der angegebenen Funktion berechnet.

- `DropDuplicates` – Ein DropDuplicates-Objekt.

  Gibt eine Transformation an, die Zeilen mit sich wiederholenden Daten aus einem Datensatz entfernt.

- `GovernedCatalogTarget` – Ein GovernedCatalogTarget-Objekt.

  Gibt ein Datenziel an, das in einen gesteuerten Katalog schreibt.

- `GovernedCatalogSource` – Ein GovernedCatalogSource-Objekt.

  Gibt eine Datenquelle in einem gesteuerten Datenkatalog an.

- `MicrosoftSQLServerCatalogSource` – Ein Microsoft SQLServer CatalogSource-Objekt.

  Gibt eine Microsoft SQL Server-Datenquelle im AWS Glue -Datenkatalog an.

- `MySQLCatalogSource` – Ein Meine SQLCatalog Quelle-Objekt.

  Gibt eine MySQL-Datenquelle im AWS Glue Datenkatalog an.

- `OracleSQLCatalogSource` – Ein SQLCatalogOracle-Quelle-Objekt.

  Gibt eine Oracle-Datenquelle im AWS Glue Datenkatalog an.

- `PostgreSQLCatalogSource` – Ein SQLCatalogPostgre-Quelle-Objekt.

  Gibt eine PostgresSQL-Datenquelle im Datenkatalog an AWS Glue .

- `MicrosoftSQLServerCatalogTarget` – Ein Microsoft SQLServer CatalogTarget-Objekt.

  Gibt ein Ziel an, das Microsoft SQL verwendet.

- `MySQLCatalogTarget` – Ein Mein SQLCatalog Ziel-Objekt.

  Gibt ein Ziel an, das MySQL verwendet.

- `OracleSQLCatalogTarget` – Ein Oracle SQLCatalog Target-Objekt.

  Gibt ein Ziel an, das Oracle SQL verwendet.

- `PostgreSQLCatalogTarget` – Ein SQLCatalogPostgre-Ziel-Objekt.

  Gibt ein Ziel an, das Postgres SQL verwendet.

- `DynamicTransform` – Ein DynamicTransform-Objekt.

  Gibt eine benutzerdefinierte visuelle Transformation an, die von einem Benutzer erstellt wurde.

- `EvaluateDataQuality` – Ein EvaluateDataQuality-Objekt.

  Gibt Ihre Auswertungskriterien für die Datenqualität an.

- `S3CatalogHudiSource` – Ein S3 CatalogHudiSource-Objekt.

  Gibt eine Hudi-Datenquelle an, die im Datenkatalog registriert ist. AWS Glue Die Datenquelle muss in Amazon S3 gespeichert werden.

- `CatalogHudiSource` – Ein CatalogHudiSource-Objekt.

  Gibt eine Hudi-Datenquelle an, die im AWS Glue Datenkatalog registriert ist.

- `S3HudiSource` – Ein S3 HudiSource-Objekt.

  Gibt eine Hudi-Datenquelle an, die in gespeichert ist. Amazon S3

- `S3HudiCatalogTarget` – Ein S3 HudiCatalogTarget-Objekt.

  Gibt ein Ziel an, das in eine Hudi-Datenquelle im AWS Glue Datenkatalog schreibt.

- `S3HudiDirectTarget` – Ein S3 HudiDirectTarget-Objekt.

  Gibt ein Ziel an, das in eine Hudi-Datenquelle in schreibt. Amazon S3

- `S3CatalogDeltaSource` – Ein S3 CatalogDeltaSource-Objekt.

  Gibt eine Delta Lake-Datenquelle an, die im AWS Glue Datenkatalog registriert ist. Die Datenquelle muss in gespeichert werden Amazon S3.

- `CatalogDeltaSource` – Ein CatalogDeltaSource-Objekt.

  Gibt eine Delta Lake-Datenquelle an, die im AWS Glue Datenkatalog registriert ist.

- `S3DeltaSource` – Ein S3 DeltaSource-Objekt.

  Gibt eine Delta Lake-Datenquelle an, die in gespeichert ist Amazon S3.

- `S3DeltaCatalogTarget` – Ein S3 DeltaCatalogTarget-Objekt.

  Gibt ein Ziel an, das in eine Delta Lake-Datenquelle im AWS Glue Datenkatalog schreibt.

- `S3DeltaDirectTarget` – Ein S3 DeltaDirectTarget-Objekt.

  Gibt ein Ziel an, das in eine Delta Lake-Datenquelle in schreibt Amazon S3.

- `AmazonRedshiftSource` – Ein AmazonRedshiftSource-Objekt.

  Gibt ein Ziel an, das in eine Datenquelle in Amazon Redshift schreibt.

- `AmazonRedshiftTarget` – Ein AmazonRedshiftTarget-Objekt.

  Gibt ein Ziel an, das in ein Datenziel in Amazon Redshift schreibt.

- `EvaluateDataQualityMultiFrame` – Ein EvaluateDataQualityMultiFrame-Objekt.

  Gibt Ihre Auswertungskriterien für die Datenqualität an. Ermöglicht mehrere Eingabedaten und gibt eine Sammlung von Dynamic Frames zurück.

- `Recipe` – Ein Rezept-Objekt.

  Gibt einen AWS Glue DataBrew Rezeptknoten an.

- `SnowflakeSource` – Ein SnowflakeSource-Objekt.

  Gibt eine Snowflake-Datenquelle an.

- `SnowflakeTarget` – Ein SnowflakeTarget-Objekt.

  Gibt ein Ziel an, das in eine Snowflake-Datenquelle schreibt.

- `ConnectorDataSource` – Ein ConnectorDataSource-Objekt.

  Gibt eine Quelle an, die mit Standardverbindungsoptionen generiert wurde.

- `ConnectorDataTarget` – Ein ConnectorDataTarget-Objekt.

  Gibt ein Ziel an, das mit Standardverbindungsoptionen generiert wurde.

JDBCConnectorStruktur der Optionen

Zusätzliche Verbindungsoptionen für den Konnektor.

Felder

- `FilterPredicate` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Zusätzliche Bedingungsklausel zum Filtern von Daten aus der Quelle. Zum Beispiel:

```
BillingCity='Mountain View'
```

Wenn Sie eine Abfrage anstelle eines Tabellennamens verwenden, sollten Sie überprüfen, ob die Abfrage mit dem angegebenen `filterPredicate` funktioniert.

- `PartitionColumn` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name einer Ganzzahlspalte, die für die Partitionierung verwendet wird. Diese Option funktioniert nur, wenn sie in `lowerBound`, `upperBound` und `numPartitions` enthalten ist. Diese Option funktioniert auf die gleiche Weise wie im Spark SQL JDBC Reader.

- `LowerBound` – Zahl (lang), nicht mehr als Keine.

  Der Mindestwert von `partitionColumn`, der verwendet wird, um Partitionsschritte festzulegen.

- `UpperBound` – Zahl (lang), nicht mehr als Keine.

  Der Maximalwert von `partitionColumn`, der verwendet wird, um Partitionsschritte festzulegen.

- `NumPartitions` – Zahl (lang), nicht mehr als Keine.

  Die Anzahl an Partitionen. Dieser Wert, zusammen mit `lowerBound` (inklusive) und `upperBound` (exklusiv), bilden Partitionsschritte für generierte `WHERE`-Klauselausdrücke, die verwendet werden, um die `partitionColumn` aufzuteilen.

- `JobBookmarkKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Der Name der Auftrags-Lesezeichenschlüssel, nach denen sortiert werden soll.

- `JobBookmarkKeysSortOrder` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Legt eine auf- oder absteigende Sortierreihenfolge fest.

- `DataTypeMapping` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (zulässige Werte: `ARRAY` | `BIGINT` | `BINARY` | `BIT` | `BLOB` | `BOOLEAN` | `CHAR` | `CLOB` | `DATALINK` | `DATE` | `DECIMAL` | `DISTINCT` | `DOUBLE` | `FLOAT` | `INTEGER` | `JAVA_OBJECT` | `LONGNVARCHAR` | `LONGVARBINARY` | `LONGVARCHAR` | `NCHAR` | `NCLOB` | `NULL` | `NUMERIC` | `NVARCHAR` | `OTHER` | `REAL` | `REF` | `REF_CURSOR` | `ROWID` | `SMALLINT` | `SQLXML` | `STRUCT` | `TIME` | `TIME_WITH_TIMEZONE` | `TIMESTAMP` | `TIMESTAMP_WITH_TIMEZONE` | `TINYINT` | `VARBINARY` | `VARCHAR`).

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (zulässige Werte: `DATE` | `STRING` | `TIMESTAMP` | `INT` | `FLOAT` | `LONG` | `BIGDECIMAL` | `BYTE` | `SHORT` | `DOUBLE`).

Benutzerdefiniertes Datentyp-Mapping, das ein Mapping aus einem JDBC-Datentyp auf einen AWS Glue -Datentyp durchführt. Die Option `"dataTypeMapping":{"FLOAT":"STRING"}` ordnet beispielsweise Datenfelder vom Typ JDBC dem `String` Typ Java `FLOAT` zu, indem sie die `ResultSet.getString()` Methode des Treibers aufruft, und verwendet sie, um den AWS Glue Datensatz zu erstellen. Das `ResultSet`-Objekt wird von jedem Treiber implementiert, sodass das Verhalten spezifisch für den von Ihnen verwendeten Treiber ist. Informieren Sie sich in der Dokumentation für Ihren JDBC-Treiber, um zu verstehen, wie der Treiber die Konvertierungen durchführt.

StreamingDataPreviewOptions Struktur

Gibt Optionen im Zusammenhang mit der Datenvorversion zum Anzeigen einer Stichprobe Ihrer Daten an.

Felder

- `PollingTime` – Zahl (lang), mindestens 10.

  Die Abrufzeit in Millisekunden.
- `RecordPollingLimit` – Zahl (lang), mindestens 1.

  Die Begrenzung der Anzahl der befragten Datensätze.

AthenaConnectorSource Struktur

Gibt einen Konnektor zu einer Amazon Athena-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Verbindung, die dem Konnektor zugeordnet ist.
- `ConnectorName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

Der Name eines Connectors, der den Zugriff auf den Datenspeicher in AWS Glue Studio unterstützt.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Art der Verbindung, wie marketplace.athena oder custom.athena, die eine Verbindung zu einem Amazon Athena-Datenspeicher bezeichnet.
- `ConnectionTable` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name der Tabelle in der Datenquelle.
- `SchemaName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name des zu lesenden CloudWatch-Protokollgruppenstreams. Beispiel, `/aws-glue/jobs/output`.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die benutzerdefinierte Athena-Quelle an.

JDBCConnectorStruktur der Quelle

Gibt einen Konnektor zu einer JDBC-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Verbindung, die dem Konnektor zugeordnet ist.
- `ConnectorName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name eines Connectors, der den Zugriff auf den Datenspeicher in AWS Glue Studio unterstützt.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Art der Verbindung, wie marketplace.jdbc oder custom.jdbc, die eine Verbindung zu einem JDBC-Datenspeicher bezeichnet.

- `AdditionalOptions` – Ein JDBCConnectorOptionen-Objekt.

  Zusätzliche Verbindungsoptionen für den Konnektor.

- `ConnectionTable` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name der Tabelle in der Datenquelle.

- `Query` – UTF-8-Zeichenfolge, die Custom string pattern #60 entspricht.

  Die Tabelle oder SQL-Abfrage, aus der die Daten abgerufen werden. Sie können `ConnectionTable` oder `query` angeben, aber nicht beides.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die benutzerdefinierte JDBC-Quelle an.

SparkConnectorSource Struktur

Gibt einen Konnektor zu einer Apache-Spark-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.

- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Verbindung, die dem Konnektor zugeordnet ist.

- `ConnectorName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name eines Connectors, der den Zugriff auf den Datenspeicher in AWS Glue Studio unterstützt.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Art der Verbindung, wie marketplace.spark oder custom.spark, die eine Verbindung zu einem Apache-Spark-Datenspeicher bezeichnet.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Zusätzliche Verbindungsoptionen für den Konnektor.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die benutzerdefinierte Sparkquelle an.

CatalogSource Struktur

Gibt einen Datenspeicher im AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

Meine SQLCatalog Quellstruktur

Gibt eine MySQL-Datenquelle im AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

SQLCatalogPostgre-Quellstruktur

Gibt eine PostgresSQL-Datenquelle im Datenkatalog an AWS Glue .

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

Oracle-Quellstruktur SQLCatalog

Gibt eine Oracle-Datenquelle im AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

SQLServerCatalogSource Microsoft-Struktur

Gibt eine Microsoft SQL Server-Datenquelle im AWS Glue -Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

CatalogKinesisSource Struktur

Gibt eine Kinesis-Datenquelle im AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `WindowSize` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Zeitspanne für die Verarbeitung der einzelnen Batches.
- `DetectSchema` – Boolesch.

Ob das Schema aus den eingehenden Daten automatisch ermittelt werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.

- `StreamingOptions` – Ein KinesisStreamingSourceOptions-Objekt.

  Zusätzliche Optionen für die Kinesis-Streaming-Datenquelle.

- `DataPreviewOptions` – Ein StreamingDataPreviewOptions-Objekt.

  Zusätzliche Optionen für die Datenvorschau.

DirectKinesisSource Struktur

Gibt eine direkte Amazon Kinesis-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.

- `WindowSize` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Zeitspanne für die Verarbeitung der einzelnen Batches.

- `DetectSchema` – Boolesch.

  Ob das Schema aus den eingehenden Daten automatisch ermittelt werden soll.

- `StreamingOptions` – Ein KinesisStreamingSourceOptions-Objekt.

  Zusätzliche Optionen für die Kinesis-Streaming-Datenquelle.

- `DataPreviewOptions` – Ein StreamingDataPreviewOptions-Objekt.

  Zusätzliche Optionen für die Datenvorschau.

KinesisStreamingSourceOptions Struktur

Zusätzliche Optionen für die Amazon Kinesis-Streaming-Datenquelle.

Felder

- `EndpointUrl` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die URL des Kinesis-Endpunktes.
- `StreamName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name des Kinesis-Datenstroms.
- `Classification` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Eine optionale Klassifizierung.
- `Delimiter` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt das Trennzeichen an.
- `StartingPosition` – UTF-8-Zeichenfolge (zulässige Werte: `latest="LATEST"` | `trim_horizon="TRIM_HORIZON"` | `earliest="EARLIEST"` | `timestamp="TIMESTAMP"`).

  Die Ausgangsposition im Kinesis Data Stream, von dem Daten gelesen werden sollen. Die möglichen Werte sind `"latest"`, `"trim_horizon"`, `"earliest"` oder eine Zeitstempelzeichenfolge im UTC-Format im Muster `yyyy-mm-ddTHH:MM:SSZ` (wobei `Z` einen UTC-Zeitzonenversatz mit einem +/- darstellt. Beispiel: „2023-04-04T08:00:00-04:00“) . Der Standardwert ist `"latest"`.

  Hinweis: Die Verwendung eines Werts, der eine Zeitstempelzeichenfolge im UTC-Format ist, für „StartingPosition“ wird nur für AWS Glue Version 4.0 oder höher unterstützt.
- `MaxFetchTimeInMs` – Zahl (lang), nicht mehr als Keine.

  Die maximale Zeit, die der Job Executor benötigt, um Datensätze für den aktuellen Batch aus dem Kinesis-Datenstream zu lesen, angegeben in Millisekunden (ms). Innerhalb dieser Zeit können mehrere `GetRecords` API-Aufrufe getätigt werden. Der Standardwert ist `1000`.
- `MaxFetchRecordsPerShard` – Zahl (lang), nicht mehr als Keine.

  Die maximale Anzahl von Datensätzen, die pro Shard im Kinesis-Datenstrom pro Mikrobatch abgerufen werden können. Hinweis: Der Client kann dieses Limit überschreiten, wenn der Streaming-Job bereits zusätzliche Datensätze von Kinesis gelesen hat (im selben Get-Records-

Aufruf). Wenn es streng sein `MaxFetchRecordsPerShard` muss, muss es ein Vielfaches von sein. `MaxRecordPerRead` Der Standardwert ist `100000`.

- `MaxRecordPerRead` – Zahl (lang), nicht mehr als Keine.

  Die maximale Anzahl von Datensätzen, die aus dem Kinesis Data Stream in jeder getRecords-Operation abgerufen werden sollen. Der Standardwert ist `10000`.

- `AddIdleTimeBetweenReads` – Boolesch.

  Fügt eine Zeitverzögerung zwischen zwei aufeinander folgenden -Operationen ein. Der Standardwert ist `"False"`. Diese Option ist nur für AWS Glue Version 2.0 und höher konfigurierbar.

- `IdleTimeBetweenReadsInMs` – Zahl (lang), nicht mehr als Keine.

  Die minimale Zeitverzögerung zwischen zwei aufeinanderfolgenden getRecords-Operationen, angegeben in Millisekunden (ms). Der Standardwert ist `1000`. Diese Option ist nur für AWS Glue Version 2.0 und höher konfigurierbar.

- `DescribeShardInterval` – Zahl (lang), nicht mehr als Keine.

  Das minimale Zeitintervall zwischen zwei ListShards API-Aufrufen, damit Ihr Skript ein Resharding in Betracht ziehen kann. Der Standardwert ist `1s`.

- `NumRetries` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die maximale Anzahl erneuter Versuche für API-Aufrufe von Kinesis Data Streams. Der Standardwert ist `3`.

- `RetryIntervalMs` – Zahl (lang), nicht mehr als Keine.

  Die Abkühlzeit (angegeben in ms) vor dem erneuten Versuch des API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `1000`.

- `MaxRetryIntervalMs` – Zahl (lang), nicht mehr als Keine.

  Die maximale Abkühlzeit (angegeben in ms) zwischen zwei wiederholten Versuchen eines API-Aufrufs von Kinesis Data Streams. Der Standardwert ist `10000`.

- `AvoidEmptyBatches` – Boolesch.

  Vermeidet das Erstellen eines leeren Mikrobatchauftrags, indem vor dem Start des Batches im Kinesis Data Stream nach ungelesenen Daten gesucht wird. Der Standardwert ist `"False"`.

- `StreamArn` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Der Amazon-Ressourcenname (ARN) des Kinesis Data Stream.

- `RoleArn` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Amazon-Ressourcenname (ARN) der Rolle, die mit AWS Security Token Service (AWS STS) übernommen werden soll. Diese Rolle muss über Berechtigungen zum Beschreiben oder Lesen von Datensatzoperationen für den Kinesis-Datenstrom verfügen. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit `"awsSTSSessionName"`.

- `RoleSessionName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Eine Kennung für die Sitzung, die die Rolle mit AWS STS übernimmt. Sie müssen diesen Parameter verwenden, wenn Sie auf einen Datenstrom in einem anderen Konto zugreifen. Verwendet in Verbindung mit `"awsSTSRoleARN"`.

- `AddRecordTimestamp` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Wenn diese Option auf 'true' gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die die Uhrzeit angibt, zu der der entsprechende Datensatz mit dem Stream empfangen wurde. Der Standardwert von "false". Diese Option wird in AWS Glue Version 4.0 oder höher unterstützt.

- `EmitConsumerLagMetrics` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Wenn diese Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, der vom Stream empfangen wurde, und dem Zeitpunkt, AWS Glue zu dem er eingeht, ausgegeben CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird in AWS Glue Version 4.0 oder höher unterstützt.

- `StartingTimestamp` – UTF-8-Zeichenfolge.

  Der Zeitstempel des Datensatzes im Kinesis-Datenstrom, aus dem mit dem Lesen der Daten begonnen werden soll. Die möglichen Werte sind eine Zeitstempelzeichenfolge im UTC-Format des Musters `yyyy-mm-ddTHH:MM:SSZ` (wobei Z einen UTC-Zeitzonenversatz mit einem +/- darstellt. Beispiel: „2023-04-04T08:00:00+08:00“).

CatalogKafkaSource Struktur

Gibt einen Apache-Kafka-Datenspeicher im Data Catalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.

- `WindowSize` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Zeitspanne für die Verarbeitung der einzelnen Batches.

- `DetectSchema` – Boolesch.

  Ob das Schema aus den eingehenden Daten automatisch ermittelt werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.

- `StreamingOptions` – Ein KafkaStreamingSourceOptions-Objekt.

  Gibt die Streaming-Optionen an.

- `DataPreviewOptions` – Ein StreamingDataPreviewOptions-Objekt.

  Gibt Optionen im Zusammenhang mit der Datenvorversion zum Anzeigen einer Stichprobe Ihrer Daten an.

DirectKafkaSource Struktur

Gibt einen Apache-Kafka-Datenspeicher an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.

- `StreamingOptions` – Ein KafkaStreamingSourceOptions-Objekt.

  Gibt die Streaming-Optionen an.

- `WindowSize` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Zeitspanne für die Verarbeitung der einzelnen Batches.

- `DetectSchema` – Boolesch.

  Ob das Schema aus den eingehenden Daten automatisch ermittelt werden soll.

- `DataPreviewOptions` – Ein StreamingDataPreviewOptions-Objekt.

  Gibt Optionen im Zusammenhang mit der Datenvorversion zum Anzeigen einer Stichprobe Ihrer Daten an.

KafkaStreamingSourceOptions Struktur

Zusätzliche Optionen zum Streaming.

Felder

- `BootstrapServers` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Eine Liste von Bootstrap-Servern URLs, zum Beispiel als`b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Diese Option muss im API-Aufruf angegeben oder in den Tabellenmetadaten im Data Catalog definiert werden.

- `SecurityProtocol` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Das Protokoll, das für die Kommunikation mit Brokern verwendet wird. Die möglichen Werte sind `"SSL"` oder `"PLAINTEXT"`.

- `ConnectionName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name der Verbindung.

- `TopicName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Name des Themas, wie in Apache Kafka angegeben. Sie müssen mindestens einen der folgenden Aspekte angeben: `"topicName"`, `"assign"` oder `"subscribePattern"`.

- `Assign` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die spezifischen zu verbrauchenden `TopicPartitions`. Sie müssen mindestens einen der folgenden Aspekte angeben: `"topicName"`, `"assign"` oder `"subscribePattern"`.

- `SubscribePattern` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Eine Java-Regex-Zeichenfolge, die die Themenliste identifiziert, die abonniert werden soll. Sie müssen mindestens einen der folgenden Aspekte angeben: `"topicName"`, `"assign"` oder `"subscribePattern"`.

- `Classification` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Eine optionale Klassifizierung.

- `Delimiter` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt das Trennzeichen an.

- `StartingOffsets` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Ausgangsposition im Kafka-Thema, aus dem Daten gelesen werden sollen. Die möglichen Werte sind `"earliest"` oder `"latest"`. Der Standardwert ist `"latest"`.

- `EndingOffsets` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Endpunkt, wenn eine Batchabfrage beendet wird. Die möglichen Werte sind entweder `"latest"` oder eine JSON-Zeichenfolge, die einen Offset für das Ende jeder `TopicPartition` angibt.

- `PollTimeoutMs` – Zahl (lang), nicht mehr als Keine.

  Das Timeout in Millisekunden, um Daten von Kafka in Spark-Auftragsausführungen abzufragen. Der Standardwert ist `512`.

- `NumRetries` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Anzahl, wie oft erneute Versuche durchgeführt werden sollen, bevor Kafka-Offsets nicht abgerufen werden. Der Standardwert ist `3`.

- `RetryIntervalMs` – Zahl (lang), nicht mehr als Keine.

  Die Wartezeit in Millisekunden, bevor Sie erneut versuchen, Kafka-Offsets abzurufen. Der Standardwert ist `10`.

- `MaxOffsetsPerTrigger` – Zahl (lang), nicht mehr als Keine.

  Die Ratengrenze für die maximale Anzahl von Offsets, die pro Triggerintervall verarbeitet werden. Die angegebene Gesamtzahl der Offsets wird proportional auf `topicPartitions`von verschiedenen Volumes aufgeteilt. Der Standardwert ist null, was bedeutet, dass der Verbraucher alle Offsets bis zum bekannten letzten Offset liest.

- `MinPartitions` – Zahl (Ganzzahl), nicht mehr als Keine.

Die gewünschte Mindestanzahl an Partitionen, die von Kafka gelesen werden sollen. Der Standardwert ist null, was bedeutet, dass die Anzahl der Spark-Partitionen gleich der Anzahl der Kafka-Partitionen ist.

- `IncludeHeaders` – Boolesch.

  Ob die Kafka-Header eingeschlossen werden sollen. Wenn die Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „glue_streaming_kafka_headers“ mit dem Typ `Array[Struct(key: String, value: String)]`. Der Standardwert ist „false“. Diese Option ist nur in AWS Glue Version 3.0 oder höher verfügbar.

- `AddRecordTimestamp` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Wenn diese Option auf „true“ gesetzt ist, enthält die Datenausgabe eine zusätzliche Spalte mit dem Namen „__src_timestamp“, die den Zeitpunkt angibt, zu dem der entsprechende Datensatz beim Thema eingegangen ist. Der Standardwert von "false". Diese Option wird in AWS Glue Version 4.0 oder höher unterstützt.

- `EmitConsumerLagMetrics` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Wenn diese Option auf „true“ gesetzt ist, werden für jeden Batch die Metriken für den Zeitraum zwischen dem ältesten Datensatz, den das Thema erhalten hat, und dem Zeitpunkt, AWS Glue zu dem er eingeht, ausgegeben CloudWatch. Der Name der Metrik lautet „glue.driver.streaming“. maxConsumerLagInMs“. Der Standardwert von "false". Diese Option wird in AWS Glue Version 4.0 oder höher unterstützt.

- `StartingTimestamp` – UTF-8-Zeichenfolge.

  Der Zeitstempel des Datensatzes im Kafka-Thema, aus dem mit dem Lesen von Daten begonnen werden soll. Die möglichen Werte sind eine Zeitstempelzeichenfolge im UTC-Format des Musters `yyyy-mm-ddTHH:MM:SSZ` (wobei Z einen UTC-Zeitzonenversatz mit einem +/- darstellt. Beispiel: „2023-04-04T08:00:00+08:00“).

  Es muss nur ein `StartingTimestamp` oder `StartingOffsets` festgelegt werden.

RedshiftSource Struktur

Gibt einen Amazon Redshift-Datenspeicher an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Amazon-Redshift-Datenspeichers.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Datenbank, aus der gelesen werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die zu lesende Datenbanktabelle.

- `RedshiftTmpDir` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Amazon S3-Pfad, in dem temporäre Daten beim Kopieren aus der Datenbank bereitgestellt werden können.

- `TmpDirIAMRole` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die IAM-Rolle mit Berechtigungen.

AmazonRedshiftSource Struktur

Gibt eine Amazon-Redshift-Quelle an.

Felder

- `Name` – UTF-8-Zeichenfolge, die Custom string pattern #61 entspricht.

  Der Name der Amazon-Redshift-Quelle.

- `Data` – Ein AmazonRedshiftNodeData-Objekt.

  Gibt die Daten des Amazon-Reshift-Quellknotens an.

AmazonRedshiftNodeData Struktur

Gibt einen Amazon-Redshift-Knoten an.

Felder

- `AccessType` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Der Zugriffstyp für die Redshift-Verbindung. Dies kann eine direkte Verbindung oder eine Katalogverbindung sein.

- `SourceType` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Der Quelltyp, der angibt, ob eine bestimmte Tabelle die Quelle oder eine benutzerdefinierte Abfrage ist.

- `Connection` – Ein Option-Objekt.

  Die AWS Glue Verbindung zum Redshift-Cluster.

- `Schema` – Ein Option-Objekt.

  Der Name des Redshift-Schemas beim Arbeiten mit einer direkten Verbindung.

- `Table` – Ein Option-Objekt.

  Der Name der Redshift-Tabelle beim Arbeiten mit einer Direktverbindung.

- `CatalogDatabase` – Ein Option-Objekt.

  Der Name der AWS Glue Datenkatalogdatenbank bei der Arbeit mit einem Datenkatalog.

- `CatalogTable` – Ein Option-Objekt.

  Der Name der AWS Glue Datenkatalogtabelle bei der Arbeit mit einem Datenkatalog.

- `CatalogRedshiftSchema` – UTF-8-Zeichenfolge.

  Der Name des Redshift-Schemas bei der Arbeit mit einem Datenkatalog.

- `CatalogRedshiftTable` – UTF-8-Zeichenfolge.

  Die zu lesende Datenbanktabelle.

- `TempDir` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Amazon S3-Pfad, in dem temporäre Daten beim Kopieren aus der Datenbank bereitgestellt werden können.

- `IamRole` – Ein Option-Objekt.

Optional. Der Rollenname, der bei der Verbindung mit S3 verwendet wird. Wenn das Feld leer gelassen wird, wird standardmäßig die Rolle des Auftrags als IAM-Rolle verwendet.

- `AdvancedOptions` – Ein Array mit AmazonRedshiftAdvancedOption-Objekten.

  Optionale Werte beim Herstellen einer Verbindung zum Redshift-Cluster.

- `SampleQuery` – UTF-8-Zeichenfolge.

  Das SQL, das zum Abrufen der Daten aus einer Redshift-Quelle verwendet wird, wenn es sich um eine „Abfrage“ SourceType handelt.

- `PreAction` – UTF-8-Zeichenfolge.

  Die SQL, die vor der Ausführung eines MERGE- oder APPEND-Vorgangs mit Upsert verwendet wird.

- `PostAction` – UTF-8-Zeichenfolge.

  Die SQL, die vor der Ausführung eines MERGE- oder APPEND-Vorgangs mit Upsert verwendet wird.

- `Action` – UTF-8-Zeichenfolge.

  Gibt an, wie in einen Redshift-Cluster geschrieben wird.

- `TablePrefix` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Gibt das Präfix für eine Tabelle an.

- `Upsert` – Boolesch.

  Die auf Redshift verwendete Aktion sinkt, wenn ein APPEND-Vorgang durchgeführt wird.

- `MergeAction` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Die Aktion, mit der bestimmt wird, wie ein MERGE-Vorgang in einer Redshift-Senke behandelt wird.

- `MergeWhenMatched` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Die Aktion, mit der bestimmt wird, wie ein MERGE-Vorgang in einer Redshift-Senke gehandhabt wird, wenn ein vorhandener Datensatz mit einem neuen Datensatz übereinstimmt.

- `MergeWhenNotMatched` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

Die Aktion, mit der bestimmt wird, wie ein MERGE-Vorgang in einer Redshift-Senke behandelt wird, wenn ein vorhandener Datensatz nicht mit einem neuen Datensatz übereinstimmt.

- `MergeClause` – UTF-8-Zeichenfolge.

  Das SQL, das bei einer benutzerdefinierten Zusammenführung zum Umgang mit übereinstimmenden Datensätzen verwendet wird.
- `CrawlerConnection` – UTF-8-Zeichenfolge.

  Gibt den Namen der Verbindung an, die der verwendeten Katalogtabelle zugeordnet ist.
- `TableSchema` – Ein Array mit Option-Objekten.

  Das Array der Schemaausgabe für einen bestimmten Knoten.
- `StagingTable` – UTF-8-Zeichenfolge.

  Der Name der temporären Staging-Tabelle, die beim Ausführen eines MERGE- oder APPEND-Vorgangs mit Upsert verwendet wird.
- `SelectedColumns` – Ein Array mit Option-Objekten.

  Die Liste der Spaltennamen, die verwendet wird, um einen passenden Datensatz zu ermitteln, wenn ein MERGE- oder APPEND-Vorgang mit Upsert durchgeführt wird.

AmazonRedshiftAdvancedOption Struktur

Gibt einen optionalen Wert an, wenn eine Verbindung zum Redshift-Cluster hergestellt wird.

Felder

- `Key` – UTF-8-Zeichenfolge.

  Der Schlüssel für die zusätzliche Verbindungsoption.
- `Value` – UTF-8-Zeichenfolge.

  Der Wert für die zusätzliche Verbindungsoption.

Optionsstruktur

Gibt einen Optionswert an.

Felder

- `Value` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt den Wert der Option an.
- `Label` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Bezeichnung der Option an.
- `Description` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Beschreibung der Option an.

CatalogSource S3-Struktur

Gibt einen Amazon S3 S3-Datenspeicher im AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die zu lesende Datenbanktabelle.
- `PartitionPredicate` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Partitionen, die diese Bedingung erfüllen, werden gelöscht. Dateien innerhalb des Aufbewahrungszeitraums in diesen Partitionen werden nicht gelöscht. Festgelegt auf "" – standardmäßig auf leer festgelegt.
- `AdditionalOptions` – Ein S3 SourceAdditionalOptions-Objekt.

  Gibt zusätzliche Verbindungsoptionen an.

SourceAdditionalOptions S3-Struktur

Gibt zusätzliche Verbindungsoptionen für den Amazon S3-Datenspeicher an.

Felder

- `BoundedSize` – Zahl (lang).

  Legt die Obergrenze für die Zielgröße des Datensatzes, die verarbeitet werden, in Byte fest.
- `BoundedFiles` – Zahl (lang).

  Legt die Obergrenze für die Zielanzahl von Dateien fest, die verarbeitet werden.

CsvSource S3-Struktur

Gibt einen CSV-Datenspeicher an, der in Amazon S3 gespeichert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- `CompressionType` – UTF-8-Zeichenfolge (zulässige Werte: `gzip="GZIP"` | `bzip2="BZIP2"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).
- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Zeichenfolge, die eine JSON-Liste der auszuschließenden Glob-Muster im Unix-Stil enthält. Beispiel: „[\"**.pdf \"]“ schließt alle PDF-Dateien aus.
- `GroupSize` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Größe der Zielgruppe in Bytes. Der Standardwert wird anhand der Größe der Eingabedaten und der Größe des Clusters berechnet. Wenn es weniger als 50 000 Eingabedateien gibt, muss `"groupFiles"` auf `"inPartition"` gesetzt werden, damit dies wirksam wird.
- `GroupFiles` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Die Gruppierung von Dateien ist standardmäßig aktiviert, wenn die Eingabe mehr als 50 000 Dateien enthält. Um die Gruppierung mit weniger als 50.000 Dateien zu aktivieren, setzen Sie diesen Parameter auf "inPartition". Um die Gruppierung zu deaktivieren, wenn mehr als 50 000 Dateien vorhanden sind, setzen Sie diesen Parameter auf `"none"`.

- `Recurse` – Boolesch.

  Wenn auf true gesetzt, werden Dateien in allen Unterverzeichnissen unter den angegebenen Pfaden rekursiv gelesen.

- `MaxBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option steuert die Dauer in Millisekunden, nach der die S3-Auflistung wahrscheinlich konsistent ist. Dateien mit Änderungszeitstempeln, die innerhalb der letzten MaxBand-Millisekunden liegen, werden speziell nachverfolgt, wenn sie verwendet werden, um die eventuelle Konsistenz von JobBookmarks Amazon S3 zu gewährleisten. Die meisten Benutzer müssen diese Option nicht festlegen. Der Standardwert ist 900 000 Millisekunden oder 15 Minuten

- `MaxFilesInBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option gibt die maximale Anzahl von Dateien an, die aus den letzten maxBand Sekunden gespeichert werden sollen. Wird diese Anzahl überschritten, werden zusätzliche Dateien übersprungen und erst bei der nächsten Auftragsausführung verarbeitet.

- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Gibt zusätzliche Verbindungsoptionen an.

- `Separator` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `comma="COMMA"` | `ctrla="CTRLA"` | `pipe="PIPE"` | `semicolon="SEMICOLON"` | `tab="TAB"`).

  Gibt das Trennzeichen an. Der Standardwert ist ein Komma: „,“, es können aber alle anderen Zeichen angegeben werden.

- `Escaper` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt das Escape-Zeichen an. Diese Option wird nur beim Lesen von CSV-Dateien verwendet. Der Standardwert ist `none`. Wenn diese Option aktiviert ist, wird das unmittelbar folgende Zeichen als solches verwendet, außer einer kleinen Menge bekannter Escapes (`\n`, `\r`, `\t` und `\0`).

- `QuoteChar` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `quote="QUOTE"` | `quillemet="QUILLEMET"` | `single_quote="SINGLE_QUOTE"` | `disabled="DISABLED"`).

Gibt das Zeichen für Anführungszeichen an. Standardmäßig werden doppelte Anführungszeichen `'"'` verwendet. Setzen Sie dies auf `-1`, um Anführungszeichen generell zu deaktivieren.

- `Multiline` – Boolesch.

  Ein boolescher Wert, der angibt, ob ein einzelner Datensatz mehrere Zeilen umfassen kann. Dies kommt vor, wenn ein Feld ein Neue-Zeile-Zeichen in Anführungszeichen enthält. Sie müssen diese Option auf "true" setzen, wenn ein Datensatz mehrere Zeilen umfasst. Der Standardwert ist `False`. Dies ermöglicht eine rigorosere Dateiaufteilung während der Analyse.

- `WithHeader` – Boolesch.

  Ein boolescher Wert, der angibt, ob die erste Zeile als Kopfzeile zu behandeln ist. Der Standardwert ist `False`.

- `WriteHeader` – Boolesch.

  Ein boolescher Wert, der angibt, ob die Kopfzeile mit ausgegeben wird. Der Standardwert ist `True`.

- `SkipFirst` – Boolesch.

  Ein boolescher Wert, der angibt, ob die erste Datenzeile übersprungen wird. Der Standardwert ist `False`.

- `OptimizePerformance` – Boolesch.

  Ein boolescher Wert, der angibt, ob der erweiterte SIMD-CSV-Reader zusammen mit Apache Arrow basierten spaltenförmigen Speicherformaten verwendet werden soll. Nur in Version 3.0 verfügbar. AWS Glue

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die S3-CSV-Quelle an.

Direkte JDBCSource Struktur

Gibt die direkte JDBC-Quellverbindung an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der JDBC-Quellverbindung.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Datenbank der JDBC-Quellverbindung.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Tabelle der JDBC-Quellverbindung.

- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Verbindungsname der JDBC-Quelle.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `sqlserver` | `mysql` | `oracle` | `postgresql` | `redshift`).

  Der Verbindungstyp der JDBC-Quelle.

- `RedshiftTmpDir` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Das temporäre Verzeichnis der JDBC-Redshift-Quelle.

DirectSourceAdditionalOptions S3-Struktur

Gibt zusätzliche Verbindungsoptionen für den Amazon S3-Datenspeicher an.

Felder

- `BoundedSize` – Zahl (lang).

  Legt die Obergrenze für die Zielgröße des Datensatzes, die verarbeitet werden, in Byte fest.

- `BoundedFiles` – Zahl (lang).

  Legt die Obergrenze für die Zielanzahl von Dateien fest, die verarbeitet werden.

- `EnableSamplePath` – Boolesch.

  Legt die Option zum Aktivieren eines Beispielpfads fest.

- `SamplePath` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Wenn diese Option aktiviert ist, gibt den Beispielpfad an.

JsonSource S3-Struktur

Gibt einen JSON-Datenspeicher an, der in Amazon S3 gespeichert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- `CompressionType` – UTF-8-Zeichenfolge (zulässige Werte: `gzip="GZIP"` | `bzip2="BZIP2"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).
- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Zeichenfolge, die eine JSON-Liste der auszuschließenden Glob-Muster im Unix-Stil enthält. Beispiel: „[\"**.pdf \"]“ schließt alle PDF-Dateien aus.
- `GroupSize` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Größe der Zielgruppe in Bytes. Der Standardwert wird anhand der Größe der Eingabedaten und der Größe des Clusters berechnet. Wenn es weniger als 50 000 Eingabedateien gibt, muss `"groupFiles"` auf `"inPartition"` gesetzt werden, damit dies wirksam wird.
- `GroupFiles` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Gruppierung von Dateien ist standardmäßig aktiviert, wenn die Eingabe mehr als 50 000 Dateien enthält. Um die Gruppierung mit weniger als 50.000 Dateien zu aktivieren, setzen Sie diesen Parameter auf "inPartition". Um die Gruppierung zu deaktivieren, wenn mehr als 50 000 Dateien vorhanden sind, setzen Sie diesen Parameter auf `"none"`.
- `Recurse` – Boolesch.

  Wenn auf true gesetzt, werden Dateien in allen Unterverzeichnissen unter den angegebenen Pfaden rekursiv gelesen.
- `MaxBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option steuert die Dauer in Millisekunden, nach der die S3-Auflistung wahrscheinlich konsistent ist. Dateien mit Änderungszeitstempeln, die innerhalb der letzten MaxBand-

Millisekunden liegen, werden speziell nachverfolgt, wenn sie verwendet werden, um die eventuelle Konsistenz von JobBookmarks Amazon S3 zu gewährleisten. Die meisten Benutzer müssen diese Option nicht festlegen. Der Standardwert ist 900 000 Millisekunden oder 15 Minuten

- `MaxFilesInBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option gibt die maximale Anzahl von Dateien an, die aus den letzten maxBand Sekunden gespeichert werden sollen. Wird diese Anzahl überschritten, werden zusätzliche Dateien übersprungen und erst bei der nächsten Auftragsausführung verarbeitet.

- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Gibt zusätzliche Verbindungsoptionen an.

- `JsonPath` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Eine JsonPath Zeichenfolge, die die JSON-Daten definiert.

- `Multiline` – Boolesch.

  Ein boolescher Wert, der angibt, ob ein einzelner Datensatz mehrere Zeilen umfassen kann. Dies kommt vor, wenn ein Feld ein Neue-Zeile-Zeichen in Anführungszeichen enthält. Sie müssen diese Option auf "true" setzen, wenn ein Datensatz mehrere Zeilen umfasst. Der Standardwert ist `False`. Dies ermöglicht eine rigorosere Dateiaufteilung während der Analyse.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die S3-JSON-Quelle an.

ParquetSource S3-Struktur

Gibt einen Apache-Parquet-Datenspeicher an, der in Amazon S3 gespeichert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.

- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.

- `CompressionType` – UTF-8-Zeichenfolge (zulässige Werte: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).

- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Zeichenfolge, die eine JSON-Liste der auszuschließenden Glob-Muster im Unix-Stil enthält. Beispiel: „[\"**.pdf \"]“ schließt alle PDF-Dateien aus.

- `GroupSize` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Größe der Zielgruppe in Bytes. Der Standardwert wird anhand der Größe der Eingabedaten und der Größe des Clusters berechnet. Wenn es weniger als 50 000 Eingabedateien gibt, muss `"groupFiles"` auf `"inPartition"` gesetzt werden, damit dies wirksam wird.

- `GroupFiles` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die Gruppierung von Dateien ist standardmäßig aktiviert, wenn die Eingabe mehr als 50 000 Dateien enthält. Um die Gruppierung mit weniger als 50.000 Dateien zu aktivieren, setzen Sie diesen Parameter auf "inPartition". Um die Gruppierung zu deaktivieren, wenn mehr als 50 000 Dateien vorhanden sind, setzen Sie diesen Parameter auf `"none"`.

- `Recurse` – Boolesch.

  Wenn auf true gesetzt, werden Dateien in allen Unterverzeichnissen unter den angegebenen Pfaden rekursiv gelesen.

- `MaxBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option steuert die Dauer in Millisekunden, nach der die S3-Auflistung wahrscheinlich konsistent ist. Dateien mit Änderungszeitstempeln, die innerhalb der letzten MaxBand-Millisekunden liegen, werden speziell nachverfolgt, wenn sie verwendet werden, um die eventuelle Konsistenz von JobBookmarks Amazon S3 zu gewährleisten. Die meisten Benutzer müssen diese Option nicht festlegen. Der Standardwert ist 900 000 Millisekunden oder 15 Minuten

- `MaxFilesInBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Diese Option gibt die maximale Anzahl von Dateien an, die aus den letzten maxBand Sekunden gespeichert werden sollen. Wird diese Anzahl überschritten, werden zusätzliche Dateien übersprungen und erst bei der nächsten Auftragsausführung verarbeitet.

- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Gibt zusätzliche Verbindungsoptionen an.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die S3-Parkett-Quelle an.

DeltaSource S3-Struktur

Spezifiziert eine Delta Lake-Datenquelle, die in gespeichert ist Amazon S3.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Delta-Lake-Quelle.

- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.

- `AdditionalDeltaOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.

- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Gibt zusätzliche Optionen für den Konnektor an.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Delta-Lake-Quelle an.

CatalogDeltaSource S3-Struktur

Gibt eine Delta Lake-Datenquelle an, die im AWS Glue Datenkatalog registriert ist. Die Datenquelle muss in gespeichert werden Amazon S3.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Delta-Lake-Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.
- `AdditionalDeltaOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Delta-Lake-Quelle an.

CatalogDeltaSource Struktur

Gibt eine Delta Lake-Datenquelle an, die im AWS Glue Datenkatalog registriert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Delta-Lake-Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

- `AdditionalDeltaOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Delta-Lake-Quelle an.

HudiSource S3-Struktur

Spezifiziert eine Hudi-Datenquelle, die in Amazon S3 gespeichert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Hudi-Quelle.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Amazon-S3-Pfade, aus denen gelesen werden soll.
- `AdditionalHudiOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.
- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Gibt zusätzliche Optionen für den Konnektor an.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Hudi-Quelle an.

S3-Struktur CatalogHudiSource

Gibt eine Hudi-Datenquelle an, die im AWS Glue Datenkatalog registriert ist. Die Hudi-Datenquelle muss in gespeichert werden. Amazon S3

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Hudi-Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.
- `AdditionalHudiOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Hudi-Quelle an.

S3-Struktur ExcelSource

Gibt eine S3-Excel-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der S3-Excel-Datenquelle.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die S3-Pfade, in denen sich die Excel-Dateien befinden.

- `CompressionType` – UTF-8-Zeichenfolge (zulässige Werte: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Das für die Excel-Dateien verwendete Komprimierungsformat.

- `Exclusions` – Ein UTF-8-Zeichenfolgen-Array.

  Muster, um bestimmte Dateien oder Pfade von der Verarbeitung auszuschließen.

- `GroupSize` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Definiert die Größe von Dateigruppen für die Stapelverarbeitung.

- `GroupFiles` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt an, wie Dateien für die Verarbeitung gruppiert werden sollen.

- `Recurse` – Boolesch.

  Gibt an, ob Unterverzeichnisse rekursiv verarbeitet werden sollen.

- `MaxBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die maximale Anzahl der zu verwendenden Verarbeitungsbänder.

- `MaxFilesInBand` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die maximale Anzahl von Dateien, die in jedem Band verarbeitet werden können.

- `AdditionalOptions` – Ein S3 DirectSourceAdditionalOptions-Objekt.

  Zusätzliche Konfigurationsoptionen für die direkte S3-Quellverarbeitung.

- `NumberRows` – Zahl (lang).

  Die Anzahl der zu verarbeitenden Zeilen aus jeder Excel-Datei.

- `SkipFooter` – Zahl (Ganzzahl), nicht mehr als Keine.

  Die Anzahl der Zeilen, die am Ende jeder Excel-Datei übersprungen werden sollen.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Die AWS Glue Schemas, die auf die verarbeiteten Daten angewendet werden sollen.

CatalogHudiSource Struktur

Gibt eine Hudi-Datenquelle an, die im AWS Glue Datenkatalog registriert ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Hudi-Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.
- `AdditionalHudiOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen an.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die Hudi-Quelle an.

Dynamo-Quellstruktur DBCatalog

Gibt eine DynamoDB-Datenquelle im Datenkatalog an AWS Glue .

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

RelationalCatalogSource Struktur

Gibt eine Datenquelle für relationale Datenbank im AWS Glue -Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenquelle.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, aus der gelesen werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, aus der gelesen werden soll.

JDBCConnectorZielstruktur

Gibt ein Datenziel an, das in den Säulenspeicher von Apache Parquet in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Verbindung, die dem Konnektor zugeordnet ist.

- `ConnectionTable` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle im Datenziel.
- `ConnectorName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Einen Namen für den Konnektor, der verwendet werden wird.
- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Art der Verbindung, wie marketplace.jdbc oder custom.jdbc, die eine Verbindung zu einem JDBC-Datenziel bezeichnet.
- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Zusätzliche Verbindungsoptionen für den Konnektor.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für das JDBC-Ziel an.

SparkConnectorTarget Struktur

Gibt ein Ziel an, das einen Apache-Spark-Konnektor verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `ConnectionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

Der Name einer Verbindung für einen Apache-Spark-Konnektor.

- `ConnectorName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name eines Apache-Spark-Konnektors.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Art der Verbindung, wie marketplace.spark oder custom.spark, die eine Verbindung zu einem Apache-Spark-Datenspeicher bezeichnet.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Zusätzliche Verbindungsoptionen für den Konnektor.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für das benutzerdefinierte Spark-Ziel an.

BasicCatalogTarget Struktur

Gibt ein Ziel an, das eine AWS Glue Datenkatalogtabelle verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name Ihres Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Die Partitionsschlüssel, die verwendet werden, um Daten auf der Grundlage eines bestimmten Schlüssels oder Schlüsselsatzes auf mehrere Partitionen oder Shards zu verteilen.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Datenbank, die die Tabelle enthält, die Sie als Ziel verwenden möchten. Diese Datenbank muss bereits im Data Catalog vorhanden sein.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Tabelle, die das Schema Ihrer Ausgabedaten definiert. Diese Tabelle muss bereits im -Data Catalog vorhanden sein.

Meine SQLCatalog Zielstruktur

Gibt ein Ziel an, das MySQL verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

SQLCatalogPostgre-Target-Struktur

Gibt ein Ziel an, das Postgres SQL verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

Struktur von Oracle SQLCatalog Target

Gibt ein Ziel an, das Oracle SQL verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

SQLServerCatalogTarget Microsoft-Struktur

Gibt ein Ziel an, das Microsoft SQL verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

RedshiftTarget Struktur

Gibt ein Ziel an, das Amazon Redshift verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

- `RedshiftTmpDir` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Der Amazon S3-Pfad, in dem temporäre Daten beim Kopieren aus der Datenbank bereitgestellt werden können.

- `TmpDirIAMRole` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Die IAM-Rolle mit Berechtigungen.

- `UpsertRedshiftOptions` – Ein UpsertRedshiftTargetOptions-Objekt.

  Die Reihe von Optionen zum Konfigurieren einer Upsert-Operation beim Schreiben in ein Redshift-Ziel.

AmazonRedshiftTarget Struktur

Gibt ein Amazon-Redshift-Ziel an.

Felder

- `Name` – UTF-8-Zeichenfolge, die Custom string pattern #61 entspricht.

  Der Name des Amazon-Redshift-Ziels.

- `Data` – Ein AmazonRedshiftNodeData-Objekt.

  Gibt die Daten des Amazon-Redshift-Zielknotens an.

- `Inputs` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolgen.

  Die Knoten, die Eingaben für das Datenziel sind.

UpsertRedshiftTargetOptions Struktur

Die Optionen zum Konfigurieren einer Upsert-Operation beim Schreiben in ein Redshift-Ziel.

Felder

- `TableLocation` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der physische Standort der Redshift-Tabelle.

- `ConnectionName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Der Name der Verbindung, die zum Schreiben in Redshift verwendet werden soll.

- `UpsertKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Die Schlüssel, mit denen festgestellt wird, ob eine Aktualisierung oder ein Einfügen durchgeführt werden soll.

CatalogTarget S3-Struktur

Gibt ein Datenziel an, das mithilfe des AWS Glue Datenkatalogs in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `SchemaChangePolicy` – Ein CatalogSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

GlueParquetTarget S3-Struktur

Gibt ein Datenziel an, das in den Säulenspeicher von Apache Parquet in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Ein einziger Amazon S3-Pfad, in den geschrieben wird.
- `Compression` – UTF-8-Zeichenfolge (zulässige Werte: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).
- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Definiert die Anzahl der Zielpartitionen für Parquet-Dateien beim Schreiben in Amazon S3 mit AWS Glue.
- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

CatalogSchemaChangePolicy Struktur

Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

Felder

- `EnableUpdateCatalog` – Boolesch.

  Ob das Aktualisierungsverhalten verwendet werden soll, wenn der Crawler ein geändertes Schema findet.

- `UpdateBehavior` – UTF-8-Zeichenfolge (zulässige Werte: `UPDATE_IN_DATABASE` | `LOG`).

  Das Aktualisierungsverhalten, wenn der Crawler ein geändertes Schema findet.

DirectTarget S3-Struktur

Gibt ein Datenziel an, das an Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Ein einziger Amazon S3-Pfad, in den geschrieben wird.
- `Compression` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).
- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Gibt die Anzahl der Zielpartitionen an, wenn Daten direkt in Amazon S3 geschrieben werden.
- `Format`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `json="JSON"` | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | | `delta="DELTA"` `iceberg="ICEBERG"` `hyper="HYPER"` |`xml="XML"`).

  Gibt das Datenausgabeformat für das Ziel an.
- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

S3-Struktur HudiCatalogTarget

Gibt ein Ziel an, das in eine Hudi-Datenquelle im AWS Glue Datenkatalog schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.
- `AdditionalOptions` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.
- `SchemaChangePolicy` – Ein CatalogSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

S3-Struktur HudiDirectTarget

Gibt ein Ziel an, das in eine Hudi-Datenquelle in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Amazon-S3-Pfad Ihrer Hudi-Datenquelle, in die geschrieben werden soll.

- `Compression` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `gzip="GZIP"` | `lzo="LZO"` | `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).

- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Gibt die Anzahl der Zielpartitionen für die Verteilung von Hudi-Datensatzdateien in Amazon S3 an.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.

- `Format`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `json="JSON"` | | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` `iceberg="ICEBERG"` |`hyper="HYPER"`). `xml="XML"`

  Gibt das Datenausgabeformat für das Ziel an.

- `AdditionalOptions` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.

- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

S3-Struktur DeltaCatalogTarget

Gibt ein Ziel an, das in eine Delta Lake-Datenquelle im AWS Glue Datenkatalog schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.

- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.

- `SchemaChangePolicy` – Ein CatalogSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

DeltaDirectTarget S3-Struktur

Gibt ein Ziel an, das in eine Delta Lake-Datenquelle in schreibt Amazon S3.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Amazon-S3-Pfad Ihrer Delta-Lake-Datenquelle, in die geschrieben werden soll.
- `Compression` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Gibt an, wie die Daten komprimiert werden. Dies ist in der Regel nicht notwendig, wenn die Daten eine Standard-Dateierweiterung haben. Mögliche Werte sind `"gzip"` und `"bzip"`).
- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Gibt die Anzahl der Zielpartitionen für die Verteilung von Delta Lake-Datensatzdateien in Amazon S3 an.
- `Format`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `json="JSON"` | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | | `delta="DELTA"` `iceberg="ICEBERG"` `hyper="HYPER"` |`xml="XML"`).

  Gibt das Datenausgabeformat für das Ziel an.
- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Verbindungsoptionen für den Konnektor an.
- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

S3-Struktur HyperDirectTarget

Gibt ein HyperDirect Datenziel an, das in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Die eindeutige Kennung für den HyperDirect Zielknoten.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Gibt die Eingabequelle für das HyperDirect Ziel an.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Definiert die Partitionierungsstrategie für die Ausgabedaten.
- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der S3-Speicherort, an den die Ausgabedaten geschrieben werden.
- `Compression` – UTF-8-Zeichenfolge (zulässige Werte: `uncompressed="UNCOMPRESSED"`).

  Der Komprimierungstyp, der auf die Ausgabedaten angewendet werden soll.
- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Definiert, wie Schemaänderungen bei Schreibvorgängen behandelt werden.

IcebergDirectTarget S3-Struktur

Gibt ein Ziel an, das in eine Iceberg-Datenquelle in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Gibt den eindeutigen Bezeichner für den Iceberg-Zielknoten in Ihrer Datenpipeline an.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Definiert die einzige Eingabequelle, die Daten für dieses Iceberg-Ziel bereitstellt.

- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die Spalten an, die zur Partitionierung der Iceberg-Tabellendaten in S3 verwendet werden.
- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Definiert den S3-Speicherort, an dem die Iceberg-Tabellendaten gespeichert werden.
- `Format`— Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `json="JSON"` | | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` `iceberg="ICEBERG"` |`hyper="HYPER"`). `xml="XML"`

  Gibt das Dateiformat an, das zum Speichern von Iceberg-Tabellendaten verwendet wird (z. B. Parquet, ORC).
- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Stellt zusätzliche Konfigurationsoptionen für die Anpassung des Verhaltens von Iceberg-Tabellen bereit.
- `SchemaChangePolicy` – Ein DirectSchemaChangePolicy-Objekt.

  Definiert, wie Schemaänderungen beim Schreiben von Daten in die Iceberg-Tabelle behandelt werden.
- `Compression` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `gzip="GZIP"` | `lzo="LZO"` | `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Gibt den Komprimierungscodec an, der für Iceberg-Tabellendateien in S3 verwendet wird.
- `NumberTargetPartitions` – UTF-8-Zeichenfolge.

  Legt die Anzahl der Zielpartitionen für die Verteilung von Iceberg-Tabellendateien in S3 fest.

DirectSchemaChangePolicy Struktur

Eine Richtlinie, in der die Aktualisierungsverhaltensweisen für den Crawler festgelegt sind.

Felder

- `EnableUpdateCatalog` – Boolesch.

Ob das Aktualisierungsverhalten verwendet werden soll, wenn der Crawler ein geändertes Schema findet.

- `UpdateBehavior` – UTF-8-Zeichenfolge (zulässige Werte: `UPDATE_IN_DATABASE` | `LOG`).

  Das Aktualisierungsverhalten, wenn der Crawler ein geändertes Schema findet.

- `Table` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Tabelle in der Datenbank an, für die die Schemaänderungsrichtlinie gilt.

- `Database` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Datenbank an, für die die Schemaänderungsrichtlinie gilt.

ApplyMapping Struktur

Gibt eine Transformation an, die Dateneigenschaftsschlüssel in der Datenquelle den Dateneigenschaftsschlüsseln im Datenziel zuordnet. Sie können Schlüssel umbenennen, die Datentypen für Schlüssel ändern und die Schlüssel auswählen, die aus dem Datensatz gelöscht werden sollen.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.

- `Mapping` – Erforderlich: Ein Array mit Mapping-Objekten.

  Gibt die Zuordnung der Dateneigenschaftsschlüssel in der Datenquelle den Dateneigenschaftsschlüsseln im Datenziel an.

Mapping-Struktur

Gibt die Zuordnung von Dateneigenschaftsschlüsseln an.

Felder

- `ToKey` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Was der Name der Spalte nach dem Apply-Mapping sein soll. Kann gleich sein wie `FromPath`.

- `FromPath` – Ein UTF-8-Zeichenfolgen-Array.

  Die Tabelle oder Spalte, die geändert werden soll.

- `FromType` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Typ der zu ändernden Daten.

- `ToType` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Der Datentyp, zu dem die Daten geändert werden sollen.

- `Dropped` – Boolesch.

  Wenn „true“, wird die Spalte entfernt.

- `Children` – Ein Array mit Mapping-Objekten.

  Gilt nur für verschachtelte Datenstrukturen. Wenn Sie die übergeordnete Struktur, aber auch eines ihrer untergeordneten Elemente ändern möchten, können Sie diese Datenstruktur ausfüllen. Es ist ebenfalls `Mapping`, aber sein `FromPath` wird der übergeordnete `FromPath` mit dem `FromPath` aus dieser Struktur an.

  Nehmen wir für den untergeordneten Teil an, Sie haben die Struktur:

  ```
  { "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
  "Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
  "inner", "ToType": "Double", "Dropped": false, }] }
  ```

  Sie können ein `Mapping` angeben, das wie folgt aussieht:

  ```
  { "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
  "Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
  "inner", "ToType": "Double", "Dropped": false, }] }
  ```

SelectFields Struktur

Gibt eine Transformation an, die die Dateneigenschaftsschlüssel auswählt, die Sie beibehalten möchten.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur.

DropFields Struktur

Gibt eine Transformation an, die die Dateneigenschaftsschlüssel auswählt, die Sie löschen möchten.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur.

RenameField Struktur

Gibt eine Transformation an, die einen einzelnen Dateneigenschaftsschlüssel umbenennt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.

- `SourcePath` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur für die Quelldaten.

- `TargetPath` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur für die Zieldaten.

Spigot-Struktur

Gibt eine Transformation an, die Beispiele der Daten in einen Amazon S3-Bucket schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.

- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Ein Pfad in Amazon S3, in dem die Transformation eine Teilmenge von Akten aus dem Datensatz in eine JSON-Datei in einem Amazon-S3-Bucket schreibt.

- `Topk` – Zahl (Ganzzahl), nicht mehr als 100.

  Gibt eine Reihe von Datensätzen an, die ab dem Anfang des Datensatzes geschrieben werden sollen.

- `Prob` – Zahl (Double), nicht mehr als 1.

Die Wahrscheinlichkeit (ein Dezimalwert mit einem Höchstwert von 1), einen bestimmten Datensatz auszuwählen. Der Wert 1 gibt an, dass jede aus dem Datensatz gelesene Zeile in die Beispielausgabe aufgenommen werden sollte.

Join-Struktur

Gibt eine Transformation an, die zwei Datensätze mit einer Vergleichsphrase für die angegebenen Dateneigenschaftsschlüssel zu einem Datensatz zusammenführt. Sie können innere, äußere, linke, rechte, linke Hälfte und linke Anti-Joins verwenden.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 2 und nicht mehr als 2 Zeichenfolgen.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `JoinType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `equijoin="EQUIJOIN"` | `left="LEFT"` | `right="RIGHT"` | `outer="OUTER"` | `leftsemi="LEFT_SEMI"` | `leftanti="LEFT_ANTI"`).

  Gibt den Typ des Join an, der für die Datensätze ausgeführt werden soll.
- `Columns` – Erforderlich: Ein Array mit JoinColumn-Objekten, nicht weniger als 2 und nicht mehr als 2 Strukturen.

  Eine Liste der beiden zu verbindenden Spalten.

JoinColumn Struktur

Gibt eine Spalte an, die verbunden werden soll.

Felder

- `From` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die Spalte, die verbunden werden soll.

- `Keys` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Der Schlüssel der zu verbindenden Spalte.

SplitFields Struktur

Gibt eine Transformation an, die Dateneigenschaftsschlüssel in zwei te`DynamicFrames` aufteilt. Die Ausgabe ist eine Sammlung von `DynamicFrames`: Einer mit ausgewählten Dateneigenschaftsschlüsseln und einer mit den übrigen Dateneigenschaftsschlüsseln.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Paths` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur.

SelectFromCollection Struktur

Gibt eine Transformation an, die einen `DynamicFrame` aus einer Reihe von `DynamicFrames` auswählt. Die Ausgabe ist das ausgewählte `DynamicFrame`.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Index` – Erforderlich: Zahl (Ganzzahl), nicht mehr als Keine.

Der Index für die DynamicFrame , die ausgewählt werden sollen.

FillMissingValues Struktur

Gibt eine Transformation an, bei der Akten im Datensatz gesucht werden können, die fehlende Werte aufweisen, und die ein neues Feld mit einem durch Imputation bestimmten Wert hinzufügt. Mit dem Eingabedatensatz wird das Modell zum Machine Learning (ML) trainiert, das den fehlenden Wert bestimmt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `ImputedPath` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur für den Datensatz, der unterstellt wird.
- `FilledPath` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Ein JSON-Pfad zu einer Variablen in der Datenstruktur für den Datensatz, der ausgefüllt wird.

Filter-Struktur

Gibt eine Transformation an, die einen Datensatz basierend auf einer Filterbedingung in zwei Teile teilt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

Die durch ihre Knotennamen identifizierten Dateneingaben.

- `LogicalOperator` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `AND` | `OR`).

  Der Operator, der verwendet wurde, um Zeilen zu filtern, indem er den Schlüsselwert mit einem bestimmten Wert vergleicht.

- `Filters` – Erforderlich: Ein Array mit FilterExpression-Objekten.

  Gibt einen Filterausdruck an.

FilterExpression Struktur

Gibt einen Filterausdruck an.

Felder

- `Operation` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `EQ` | `LT` | `GT` | `LTE` | `GTE` | `REGEX` | `ISNULL`).

  Der Typ des Vorgangs, der im Ausdruck ausgeführt werden soll.

- `Negated` – Boolesch.

  Ob der Ausdruck negiert werden soll.

- `Values` – Erforderlich: Ein Array mit FilterValue-Objekten.

  Eine Liste von Filterwerten.

FilterValue Struktur

Stellt einen einzelnen Eintrag in der Liste von Werten für ein `FilterExpression` dar.

Felder

- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `COLUMNEXTRACTED` | `CONSTANT`).

  Der Typ des Filterwerts.

- `Value` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Der Wert, der zugeordnet werden soll.

CustomCode Struktur

Gibt eine Transformation an, die benutzerdefinierten Code verwendet, den Sie zur Durchführung der Datentransformation bereitstellen. Die Ausgabe ist eine Sammlung von DynamicFrames.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich:Ein Array mit UTF-8-Zeichenfolgen, mindestens 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Code` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #52 übereinstimmt.

  Der benutzerdefinierte Code, der zur Durchführung der Datentransformation verwendet wird.
- `ClassName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name, der für die benutzerdefinierte Code-Knotenklasse definiert wurde.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die benutzerdefinierte Codetransformation an.

SparkSQL-Struktur

Gibt eine Transformation an, bei der Sie eine SQL-Abfrage mit der Spark SQL-Syntax eingeben, um die Daten zu transformieren. Die Ausgabe ist ein einzelner `DynamicFrame`.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich:Ein Array mit UTF-8-Zeichenfolgen, mindestens 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben. Sie können jedem Eingabeknoten einen Tabellennamen zuordnen, der in der SQL-Abfrage verwendet werden soll. Der von Ihnen gewählte Name muss den Benennungsbeschränkungen von Spark SQL entsprechen.

- `SqlQuery` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #60 übereinstimmt.

  Eine SQL-Abfrage, die die Spark SQL-Syntax verwenden und einen einzelnen Datensatz zurückgeben muss.

- `SqlAliases` – Erforderlich: Ein Array mit SqlAlias-Objekten.

  Eine Liste von Aliasen. Mit einem Alias können Sie angeben, welcher Namen in der SQL für eine bestimmte Eingabe verwendet werden soll. Sie haben beispielsweise eine Datenquelle mit dem Namen ""MyDataSource. Wenn Sie as und `From Alias` as MyDataSource angeben SqlName, können Sie in Ihrem SQL Folgendes tun:

  ```
  select * from SqlName
  ```

  und das bezieht Daten von MyDataSource.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die sparkSQL-Transformation an.

SqlAlias Struktur

Stellt einen einzelnen Eintrag in der Liste von Werten für `SqlAliases` dar.

Felder

- `From` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #58 übereinstimmt.

  Eine Tabelle oder eine Spalte in einer Tabelle.

- `Alias` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Ein temporärer Name, der einer Tabelle oder einer Spalte in einer Tabelle gegeben wird.

DropNullFields Struktur

Gibt eine Transformation an, die Spalten aus dem Datensatz entfernt, wenn alle Werte in der Spalte „null“ sind. Standardmäßig erkennt AWS Glue Studio Null-Objekte, aber einige Werte wie leere Zeichenketten, Zeichenketten, die „Null“ sind, Ganzzahlen von -1 oder andere Platzhalter wie Nullen, werden nicht automatisch als Nullen erkannt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.

- `NullCheckBoxList` – Ein NullCheckBoxList-Objekt.

  Eine Struktur, die angibt, ob bestimmte Werte als zu entfernende Nullwerte erkannt werden.

- `NullTextList` – Ein Array mit NullValueField-Objekten, nicht mehr als 50 Strukturen.

  Eine Struktur, die eine Liste von NullValueField Strukturen angibt, die einen benutzerdefinierten Nullwert wie Null oder einen anderen Wert darstellen, der als Null-Platzhalter verwendet wird, der nur für den Datensatz gilt.

  Die `DropNullFields`-Transformation entfernt benutzerdefinierte Nullwerte nur, wenn sowohl der Wert des Nullplatzhalters als auch der Datentyp mit den Daten übereinstimmen.

NullCheckBoxList Struktur

Gibt an, ob bestimmte Werte als zu entfernende Nullwerte erkannt werden.

Felder

- `IsEmpty` – Boolesch.

  Gibt an, dass eine leere Zeichenfolge als Nullwert angesehen wird.

- `IsNullString` – Boolesch.

  Gibt an, dass ein Wert, der das Wort 'null' ausgibt, als Nullwert betrachtet wird.

- `IsNegOne` – Boolesch.

  Gibt an, dass ein Ganzzahlwert von -1 als Nullwert angesehen wird.

NullValueField Struktur

Stellt einen benutzerdefinierten Nullwert wie Null oder einen anderen Wert dar, der als für den Datensatz eindeutigen Null-Platzhalter verwendet wird.

Felder

- `Value` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Wert des Null-Platzhalters.
- `Datatype` – Erforderlich: Ein Datatype-Objekt.

  Der Datentyp des Wertes.

Datatype-Struktur

Eine Struktur, die den Datentyp des Wertes darstellt.

Felder

- `Id` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #58 übereinstimmt.

  Der Datentyp des Wertes.
- `Label` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #58 übereinstimmt.

  Ein Label, das dem Datentyp zugewiesen ist.

Merge-Struktur

Gibt eine Transformation an, die ein `DynamicFrame` mit einem Staging-`DynamicFrame` basierend auf den angegebenen Primärschlüsseln zusammenführt, um Datensätze zu identifizieren. Doppelte Datensätze (Datensätze mit denselben Primärschlüsseln) werden nicht dedupliziert.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 2 und nicht mehr als 2 Zeichenfolgen.

Die durch ihre Knotennamen identifizierten Dateneingaben.

- `Source` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #58 übereinstimmt.

  Der `DynamicFrame` der Quelle, der mit einem Staging-`DynamicFrame` zusammengeführt werden wird.

- `PrimaryKeys` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die Liste der Primärschlüsselfelder, die Datensätze aus den Quell- und dynamischen Staging-Frames abgleichen.

Union-Struktur

Gibt eine Transformation an, die die Zeilen aus zwei oder mehr Datensätzen zu einem einzigen Ergebnis kombiniert.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 2 und nicht mehr als 2 Zeichenfolgen.

  Die Knoten-ID gibt die Transformation ein.

- `UnionType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `ALL` | `DISTINCT`).

  Gibt den Typ der Union-Transformation an.

  Geben Sie `ALL` an, dass alle Zeilen aus Datenquellen mit dem Ergebnis verknüpft werden sollen DynamicFrame. Die resultierende Verbindung entfernt keine doppelten Zeilen.

  Geben Sie `DISTINCT` an, ob doppelte Zeilen im Ergebnis entfernt werden sollen DynamicFrame.

PIIDetection Struktur

Gibt eine Transformation an, die PII-Daten identifiziert, entfernt oder maskiert.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten-ID gibt die Transformation ein.

- `PiiType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `RowAudit` | `RowMasking` | `ColumnAudit` | `ColumnMasking`).

  Gibt die Art der PIIDetection Transformation an.

- `EntityTypesToDetect` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Gibt die Typen von Entitäten an, die die PIIDetection Transformation als PII-Daten identifizieren wird.

  Zu den Elementen des PII-Typs gehören: PERSON_NAME, DATE, USA_SNN, EMAIL, USA_ITIN, USA_PASSPORT_NUMBER, PHONE_NUMBER, BANK_ACCOUNT, IP_ADDRESS, MAC_ADDRESS, USA_CPT_CODE, USA_HCPCS_CODE, USA_NATIONAL_DRUG_CODE, USA_MEDICARE_BENEFICIARY_IDENTIFIER, USA_HEALTH_INSURANCE_CLAIM_NUMBER,CREDIT_CARD,USA_NATIONAL_PROVIDER_IDENTIFI

- `OutputColumnName` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt den Namen der Ausgabespalte an, der jeden in dieser Zeile erkannten Entitätstyp enthält.

- `SampleFraction` – Zahl (Double), nicht mehr als 1.

  Gibt den Bruchteil der Daten an, die beim Scannen nach PII-Entitäten erfasst werden sollen.

- `ThresholdFraction` – Zahl (Double), nicht mehr als 1.

  Gibt den Bruchteil der Daten an, der erfüllt sein muss, damit eine Spalte als PII-Daten identifiziert werden kann.

- `MaskValue` – UTF-8-Zeichenfolge, nicht mehr als 256 Bytes lang, passend zum Custom string pattern #56.

  Gibt den Wert an, der die erkannte Entität ersetzt.

Aggregierte Struktur

Gibt eine Transformation an, die Zeilen nach ausgewählten Feldern gruppiert und den aggregierten Wert nach der angegebenen Funktion berechnet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Gibt die Felder und Zeilen an, die als Eingaben für die Aggregat-Transformation verwendet werden sollen.
- `Groups` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Gibt die Felder an, nach denen gruppiert werden sollen.
- `Aggs` – Erforderlich: Ein Array mit AggregateOperation-Objekten, nicht weniger als 1 und nicht mehr als 30 Strukturen.

  Gibt die Aggregatfunktionen an, die für bestimmte Felder ausgeführt werden sollen.

DropDuplicates Struktur

Gibt eine Transformation an, die Zeilen mit sich wiederholenden Daten aus einem Datensatz entfernt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Transform-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die durch ihre Knotennamen identifizierten Dateneingaben.
- `Columns` – Ein UTF-8-Zeichenfolgen-Array.

  Der Name der Spalten, die bei Wiederholung zusammengeführt oder entfernt werden sollen.

GovernedCatalogTarget Struktur

Gibt ein Datenziel an, das mithilfe des AWS Glue Datenkatalogs in Amazon S3 schreibt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datenziels.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für das Datenziel sind.
- `PartitionKeys` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die native Partitionierung mit einer Schlüsselfolge an.
- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Tabelle in der Datenbank, in die geschrieben werden soll.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Name der Datenbank, in die geschrieben wird.
- `SchemaChangePolicy` – Ein CatalogSchemaChangePolicy-Objekt.

  Eine Richtlinie, die das Aktualisierungsverhalten für den verwalteten Katalog festlegt.

GovernedCatalogSource Struktur

Gibt den Datenspeicher im verwalteten AWS Glue Datenkatalog an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Datastores.
- `Database` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

Die Datenbank, aus der gelesen werden soll.

- `Table` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die zu lesende Datenbanktabelle.

- `PartitionPredicate` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Partitionen, die diese Bedingung erfüllen, werden gelöscht. Dateien innerhalb des Aufbewahrungszeitraums in diesen Partitionen werden nicht gelöscht. Festgelegt auf "" – standardmäßig auf leer festgelegt.

- `AdditionalOptions` – Ein S3 SourceAdditionalOptions-Objekt.

  Gibt zusätzliche Verbindungsoptionen an.

AggregateOperation Struktur

Gibt den Parametersatz an, der zum Ausführen der Aggregation in der Aggregations-Transformation erforderlich ist.

Felder

- `Column` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Gibt die Spalte im Datensatz an, auf die die Aggregationsfunktion angewendet wird.

- `AggFunc` – Erforderlich: UTF-8-Zeichenfolge (gültige Werte: `avg` | `countDistinct` | `count` | `first` | `last` | `kurtosis` | `max` | `min` | `skewness` | `stddev_samp` | `stddev_pop` | `sum` | `sumDistinct` | `var_samp` | `var_pop`).

  Gibt die Aggregationsfunktion an, die angewendet werden soll.

  Zu den möglichen Aggregationsfunktionen gehören: avg countDistinct, count, first, last, kurtosis, max, min, skewness, stddev_samp, stddev_pop, sum, sumDistinct, var_samp, var_pop

GlueSchema Struktur

Gibt ein benutzerdefiniertes Schema an, wenn ein Schema nicht durch AWS Glue bestimmt werden kann.

Felder

- `Columns` – Ein Array mit GlueStudioSchemaColumn-Objekten.

  Gibt die Spaltendefinitionen an, aus denen ein AWS Glue Schema besteht.

GlueStudioSchemaColumn Struktur

Gibt eine einzelne Spalte in einer AWS Glue Schemadefinition an.

Felder

- `Name` – Erforderlich: UTF-8-String, nicht mehr als 1 024 Bytes lang, passend zum Single-line string pattern.

  Der Name der Spalte im AWS Glue Studio-Schema.
- `Type` – UTF-8-Zeichenfolge, nicht mehr als 131 072 Bytes lang, passend zum Single-line string pattern.

  Der Strukturtyp für diese Spalte im AWS Glue Studio-Schema.

GlueStudioColumn Struktur

Gibt eine einzelne Spalte in AWS Glue Studio an.

Felder

- `Key` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der Schlüssel der Spalte in AWS Glue Studio.
- `FullPath` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  TThe vollständige URL der Spalte in AWS Glue Studio.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `array="ARRAY"` | `bigint="BIGINT"` | `bigint array="BIGINT_ARRAY"` | `binary="BINARY"` | `binary array="BINARY_ARRAY"` | `boolean="BOOLEAN"` | `boolean array="BOOLEAN_ARRAY"` | `byte="BYTE"` | `byte array="BYTE_ARRAY"` | `char="CHAR"` | `char array="CHAR_ARRAY"` | `choice="CHOICE"` | `choice array="CHOICE_ARRAY"` | `date="DATE"` | `date array="DATE_ARRAY"` | `decimal="DECIMAL"` | `decimal array="DECIMAL_ARRAY"` | `double="DOUBLE"`

| double array="DOUBLE_ARRAY" | enum="ENUM" | enum array="ENUM_ARRAY" | float="FLOAT" | float array="FLOAT_ARRAY" | int="INT" | int array="INT_ARRAY" | interval="INTERVAL" | interval array="INTERVAL_ARRAY" | long="LONG" | long array="LONG_ARRAY" | object="OBJECT" | short="SHORT" | short array="SHORT_ARRAY" | smallint="SMALLINT" | smallint array="SMALLINT_ARRAY" | string="STRING" | string array="STRING_ARRAY" | timestamp="TIMESTAMP" | timestamp array="TIMESTAMP_ARRAY" | tinyint="TINYINT" | tinyint array="TINYINT_ARRAY" | varchar="VARCHAR" | varchar array="VARCHAR_ARRAY" | null="NULL" | unknown="UNKNOWN" | unknown array="UNKNOWN_ARRAY").

  TThe Typ der Spalte in AWS Glue Studio.

- `Children` – Eine Reihe von Strukturen.

  TThe untergeordnete Elemente der übergeordneten Spalte in AWS Glue Studio.

DynamicTransform Struktur

Gibt den Parametersatz an, der zum Ausführen der dynamischen Transformation erforderlich ist.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Gibt den Namen der dynamischen Transformation an.

- `TransformName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Gibt den Namen der dynamischen Transformation an, so wie er im Visual Editor von AWS Glue Studio angezeigt wird.

- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Gibt die erforderlichen Eingaben für die dynamische Transformation an.

- `Parameters` – Ein Array mit TransformConfigParameter-Objekten.

  Gibt die Parameter der dynamischen Transformation an.

- `FunctionName` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

Gibt den Namen der Funktion der dynamischen Transformation an.

- `Path` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Gibt den Pfad der Quell- und Konfigurationsdateien der dynamischen Transformation an.

- `Version` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Dieses Feld wird nicht verwendet und wird in zukünftigen Versionen veraltet sein.

- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für die dynamische Transformation an.

TransformConfigParameter Struktur

Gibt die Parameter in der Konfigurationsdatei der dynamischen Transformation an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Gibt den Namen des Parameters in der Konfigurationsdatei der dynamischen Transformation an.

- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Gibt den Parametertyp in der Konfigurationsdatei der dynamischen Transformation an.

- `ValidationRule` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Validierungsregel in der Konfigurationsdatei der dynamischen Transformation an.

- `ValidationMessage` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt die Validierungsnachricht in der Konfigurationsdatei der dynamischen Transformation an.

- `Value` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt den Wert des Parameters in der Konfigurationsdatei der dynamischen Transformation an.

- `ListType` – UTF-8-Zeichenfolge (zulässige Werte: `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Gibt den Listentyp des Parameters in der Konfigurationsdatei der dynamischen Transformation an.

- `IsOptional` – Boolesch.

Gibt an, ob der Parameter in der Konfigurationsdatei der dynamischen Transformation optional ist oder nicht.

EvaluateDataQuality Struktur

Gibt Ihre Auswertungskriterien für die Datenqualität an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenqualitätsbewertung.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Eingaben Ihrer Datenqualitätsbewertung.
- `Ruleset` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang, passend zum Custom string pattern #57.

  Der Regelsatz für Ihre Bewertung der Datenqualität.
- `Output` – UTF-8-Zeichenfolge (zulässige Werte: `PrimaryInput` | `EvaluationResults`).

  Das Ergebnis Ihrer Datenqualitätsbewertung.
- `PublishingOptions` – Ein DQResultsPublishingOptions-Objekt.

  Optionen zum Konfigurieren der Veröffentlichung Ihrer Ergebnisse.
- `StopJobOnFailureOptions` – Ein DQStopJobOnFailureOptions-Objekt.

  Optionen zum Konfigurieren, wie Ihr Auftrag angehalten wird, wenn Ihre Datenqualitätsuswertung fehlschlägt.

DQResultsPublishingOptions Struktur

Optionen zum Konfigurieren der Veröffentlichung der Ergebnisse Ihrer Datenqualitätsauswertung.

Felder

- `EvaluationContext` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

Der Kontext der Bewertung.

- `ResultsS3Prefix` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Das Amazon-S3-Präfix wurde den Ergebnissen vorangestellt.
- `CloudWatchMetricsEnabled` – Boolesch.

  Aktivieren Sie Metriken für Ihre Datenqualitätsergebnisse.
- `ResultsPublishingEnabled` – Boolesch.

  Aktivieren Sie die Veröffentlichung Ihrer Datenqualitätsergebnisse.

DQStopJobOnFailureOptions Struktur

Optionen zum Konfigurieren, wie Ihr Auftrag angehalten wird, wenn Ihre Datenqualitätsuswertung fehlschlägt.

Felder

- `StopJobOnFailureTiming` – UTF-8-Zeichenfolge (zulässige Werte: `Immediate` | `AfterDataLoad`).

  Wann Sie den Auftrag anhalten sollten, wenn Ihre Datenqualitätsbewertung fehlschlägt. Die Optionen sind Sofort oder AfterDataLoad.

EvaluateDataQualityMultiFrame Struktur

Gibt Ihre Auswertungskriterien für die Datenqualität an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Datenqualitätsbewertung.
- `Inputs` – Erforderlich:Ein Array mit UTF-8-Zeichenfolgen, mindestens 1 Zeichenfolge.

  Die Eingaben Ihrer Datenqualitätsbewertung. Die erste Eingabe in dieser Liste ist die primäre Datenquelle.
- `AdditionalDataSources` – Ein Map-Array von Schlüssel-Wert-Paaren.

Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #61 entspricht.

Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

Die Aliase aller Datenquellen mit Ausnahme der primären.

- `Ruleset` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang, passend zum Custom string pattern #57.

  Der Regelsatz für Ihre Bewertung der Datenqualität.
- `PublishingOptions` – Ein DQResultsPublishingOptions-Objekt.

  Optionen zum Konfigurieren der Veröffentlichung Ihrer Ergebnisse.
- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge (gültige Werte: `performanceTuning.caching="CacheOption"` | `observations.scope="ObservationsOption"`).

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Optionen zum Konfigurieren des Laufzeitverhaltens der Transformation.
- `StopJobOnFailureOptions` – Ein DQStopJobOnFailureOptions-Objekt.

  Optionen zum Konfigurieren, wie Ihr Auftrag angehalten wird, wenn Ihre Datenqualitätsuswertung fehlschlägt.

Struktur des Rezepts

Ein AWS Glue Studio-Knoten, der ein AWS Glue DataBrew Rezept in AWS Glue Jobs verwendet.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des AWS Glue Studio-Knotens.
- `Inputs` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolge.

  Die Knoten, die Eingaben für den Rezeptknoten sind, identifiziert durch die ID.

- `RecipeReference` – Ein RecipeReference-Objekt.

  Ein Verweis auf das vom Knoten verwendete DataBrew Rezept.

- `RecipeSteps` – Ein Array mit RecipeStep-Objekten.

  Transformationsschritte, die im Rezeptknoten verwendet werden.

RecipeReference Struktur

Ein Verweis auf ein AWS Glue DataBrew Rezept.

Felder

- `RecipeArn` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Der ARN des DataBrew Rezepts.

- `RecipeVersion` – Erforderlich: UTF-8-Zeichenkette, nicht weniger als 1 oder mehr als 16 Bytes lang.

  Die RecipeVersion des DataBrew Rezepts.

SnowflakeNodeData Struktur

Gibt die Konfiguration für Snowflake-Knoten in Studio an AWS Glue .

Felder

- `SourceType` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Gibt an, wie abgerufene Daten angegeben werden. Zulässige Werte: `"table"`, `"query"`.

- `Connection` – Ein Option-Objekt.

  Gibt eine AWS Glue Datenkatalogverbindung zu einem Snowflake-Endpunkt an.

- `Schema` – UTF-8-Zeichenfolge.

  Gibt ein Snowflake-Datenbankschema an, das Ihr Knoten verwenden soll.

- `Table` – UTF-8-Zeichenfolge.

Gibt eine Snowflake-Tabelle an, die Ihr Knoten verwenden soll.

- `Database` – UTF-8-Zeichenfolge.

  Gibt eine Snowflake-Datenbank an, die Ihr Knoten verwenden soll.

- `TempDir` – UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Derzeit nicht verwendet.

- `IamRole` – Ein Option-Objekt.

  Derzeit nicht verwendet.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die Custom string pattern #59 entspricht.

  Gibt zusätzliche Optionen an, die an den Snowflake-Konnektor übergeben werden. Wenn an anderer Stelle in diesem Knoten Optionen angegeben werden, hat dies Vorrang.

- `SampleQuery` – UTF-8-Zeichenfolge.

  Eine SQL-Zeichenfolge, die zum Abrufen von Daten mit dem `query`-Quelltyp verwendet wird.

- `PreAction` – UTF-8-Zeichenfolge.

  Eine SQL-Zeichenfolge, die ausgeführt wird, bevor der Snowflake-Konnektor seine Standardaktionen ausführt.

- `PostAction` – UTF-8-Zeichenfolge.

  Eine SQL-Zeichenfolge, die ausgeführt wird, nachdem der Snowflake-Konnektor seine Standardaktionen ausgeführt hat.

- `Action` – UTF-8-Zeichenfolge.

  Gibt an, welche Aktion beim Schreiben in eine Tabelle mit bereits vorhandenen Daten ausgeführt werden soll. Zulässige Werte: `append`, `merge`, `truncate`, `drop`.

- `Upsert` – Boolesch.

  Wird verwendet, wenn die Aktion `append` ist. Gibt das Auflösungsverhalten an, wenn bereits eine Zeile vorhanden ist. Wenn der Wert wahr ist, werden bereits vorhandene Zeilen aktualisiert. Wenn der Wert falsch ist, werden diese Zeilen eingefügt.

- `MergeAction` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Gibt eine Zusammenführungsaktion an. Zulässige Werte: `simple`, `custom`. Wenn das Zusammenführungsverhalten einfach ist, wird es durch `MergeWhenMatched` und `MergeWhenNotMatched` definiert. Falls benutzerdefiniert, durch `MergeClause` definiert.

- `MergeWhenMatched` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Gibt an, wie Datensätze, die mit bereits vorhandenen Daten übereinstimmen, beim Zusammenführen aufgelöst werden. Zulässige Werte: `update`, `delete`.

- `MergeWhenNotMatched` – UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Gibt an, wie Datensätze, die nicht mit bereits vorhandenen Daten übereinstimmen, beim Zusammenführen verarbeitet werden. Zulässige Werte: `insert`, `none`.

- `MergeClause` – UTF-8-Zeichenfolge.

  Eine SQL-Anweisung, die ein benutzerdefiniertes Zusammenführungsverhalten angibt.

- `StagingTable` – UTF-8-Zeichenfolge.

  Der Name einer Staging-Tabelle, die beim Ausführen von `merge`- oder Upsert-`append`-Aktionen verwendet wird. Daten werden in diese Tabelle geschrieben und dann durch eine generierte Folgeaktion nach `table` verschoben.

- `SelectedColumns` – Ein Array mit Option-Objekten.

  Gibt die kombinierten Spalten an, um einen Datensatz zu identifizieren, wenn Übereinstimmungen für Zusammenführungen und Upserts ermittelt werden. Eine Liste von Strukturen mit `value`-, `label`- und `description`-Schlüsseln. Jede Struktur beschreibt eine Spalte.

- `AutoPushdown` – Boolesch.

  Gibt an, ob der automatische Abfrage-Pushdown aktiviert ist. Wenn Pushdown aktiviert ist, wird bei der Ausführung einer Abfrage auf Spark ein Teil der Abfrage auf den Snowflake-Server „heruntergeschoben“, wenn dies möglich ist. Dies verbessert die Leistung einiger Abfragen.

- `TableSchema` – Ein Array mit Option-Objekten.

  Definiert das Zielschema für den Knoten manuell. Eine Liste von Strukturen mit `value`-, `label`- und `description`-Schlüsseln. Jede Struktur definiert eine Spalte.

SnowflakeSource Struktur

Gibt eine Snowflake-Datenquelle an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name der Snowflake-Datenquelle.
- `Data` – Erforderlich: Ein SnowflakeNodeData-Objekt.

  Konfiguration für die Snowflake-Datenquelle.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt benutzerdefinierte Schemas für Ihre Ausgabedaten an.

SnowflakeTarget Struktur

Gibt ein Snowflake-Ziel an.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name des Snowflake-Ziels.
- `Data` – Erforderlich: Ein SnowflakeNodeData-Objekt.

  Gibt die Daten des Snowflake-Zielknotens an.
- `Inputs` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolgen.

  Die Knoten, die Eingaben für das Datenziel sind.

ConnectorDataSource Struktur

Gibt eine Quelle an, die mit Standardverbindungsoptionen generiert wurde.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

Der Name dieses Quell-Knotens.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die`connectionType`, wie sie der zugrunde liegenden AWS Glue Bibliothek zur Verfügung gestellt wurde. Dieser Knotentyp unterstützt die folgenden Verbindungstypen:

  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`
  - `vertica`
- `Data` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Zuordnung, die Verbindungsoptionen für den Knoten angibt. Standardverbindungsoptionen für den entsprechenden Verbindungstyp finden Sie im Abschnitt Verbindungsparameter der AWS Glue Dokumentation.
- `OutputSchemas` – Ein Array mit GlueSchema-Objekten.

  Gibt das Datenschema für diese Quelle an.

ConnectorDataTarget Struktur

Gibt ein Ziel an, das mit Standardverbindungsoptionen generiert wurde.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #61 übereinstimmt.

  Der Name dieses Ziel-Knotens.

- `ConnectionType` – Erforderlich: UTF-8-Zeichenfolge, die mit der Custom string pattern #59 übereinstimmt.

  Die`connectionType`, wie sie der zugrunde liegenden AWS Glue Bibliothek zur Verfügung gestellt wurde. Dieser Knotentyp unterstützt die folgenden Verbindungstypen:

  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`
  - `vertica`
- `Data` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Zuordnung, die Verbindungsoptionen für den Knoten angibt. Standardverbindungsoptionen für den entsprechenden Verbindungstyp finden Sie im Abschnitt Verbindungsparameter der AWS Glue Dokumentation.
- `Inputs` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 1 Zeichenfolgen.

  Die Knoten, die Eingaben für das Datenziel sind.

RecipeStep Struktur

Ein Rezeptschritt, der in einem Rezeptknoten für die Datenvorbereitung in AWS Glue Studio verwendet wird.

Felder

- `Action` – Erforderlich: Ein RecipeAction-Objekt.

  Die Transformationsaktion des Rezeptschritts.
- `ConditionExpressions` – Ein Array mit ConditionExpression-Objekten.

Die Bedingungsausdrücke für den Rezeptschritt.

RecipeAction Struktur

Aktionen, die im Rezeptknoten für die Datenvorbereitung in AWS Glue Studio definiert sind.

Felder

- `Operation` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #54.

  Die Funktionsweise der Rezeptaktion.
- `Parameters` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, und entspricht dem Custom string pattern #55.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 32768 Byte lang ist.

  Die Parameter der Rezeptaktion.

ConditionExpression Struktur

Bedingungsausdruck, der im Rezeptknoten für die Datenvorbereitung in AWS Glue Studio definiert ist.

Felder

- `Condition` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #54.

  Die Bedingung des Bedingungsausdrucks.
- `Value`— UTF-8-Zeichenfolge, nicht länger als 1024 Byte.

  Der Wert des Bedingungsausdrucks.
- `TargetColumn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Die Zielspalte der Bedingungsausdrücke.

Auftrags-API

Die Jobs-API beschreibt Job-Datentypen und enthält APIs für die Arbeit mit Jobs, Jobausführungen und Triggern in AWS Glue.

Themen

- Aufträge
- Auftragsausführungen
- Auslöser

Aufträge

Die Jobs-API beschreibt die Datentypen und die API für das Erstellen, Aktualisieren, Löschen oder Anzeigen von Jobs in AWS Glue.

Datentypen

- Auftrags-Struktur
- ExecutionProperty Struktur
- NotificationProperty Struktur
- JobCommand Struktur
- ConnectionsList Struktur
- JobUpdate Struktur
- SourceControlDetails Struktur

Auftrags-Struktur

Gibt eine Auftragsdefinition an.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name, den Sie dieser Auftragsdefinition zuweisen.
- `JobMode` – UTF-8-Zeichenfolge (zulässige Werte: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

Ein Modus, der beschreibt, wie ein Job erstellt wurde. Gültige Werte für sind:

- `SCRIPT`- Der Job wurde mit dem AWS Glue Studio-Skripteditor erstellt.
- `VISUAL`- Der Job wurde mit dem Visual Editor von AWS Glue Studio erstellt.
- `NOTEBOOK`- Der Job wurde mit einem interaktiven Sitzungsnotizbuch erstellt.

Wenn das `JobMode` Feld fehlt oder Null `SCRIPT` ist, wird es als Standardwert zugewiesen.

- `JobRunQueuingEnabled` – Boolesch.

  Gibt an, ob die Warteschlange für die Auftragsausführung für diesen Job aktiviert ist.

  Der Wert true bedeutet, dass das Job Run Queuing für die Jobausführungen aktiviert ist. Wenn der Wert falsch oder nicht ausgefüllt ist, werden die Auftragsausführungen nicht in die Warteschlange aufgenommen.

  Wenn dieses Feld nicht mit dem Wert übereinstimmt, der in der Auftragsausführung festgelegt wurde, wird der Wert aus dem Feld für die Auftragsausführung verwendet.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Aufgabe.
- `LogUri` – UTF-8-Zeichenfolge.

  Dieses Feld ist für zukünftige Zwecke reserviert.
- `Role` – UTF-8-Zeichenfolge.

  Der Name oder der Amazon-Ressourcenname (ARN) der IAM-Rolle, die diesem Auftrag zugeordnet ist.
- `CreatedOn` – Zeitstempel.

  Datum und Uhrzeit der Erstellung dieser Auftragsdefinition.
- `LastModifiedOn` – Zeitstempel.

  Der letzte Zeitpunkt, zu dem diese Auftragsdefinition geändert wurde.
- `ExecutionProperty` – Ein ExecutionProperty-Objekt.

  Eine `ExecutionProperty`, die die maximale Anzahl der gleichzeitigen Ausführungen angibt, die für diesen Auftrag zulässig sind.

- `Command` – Ein JobCommand-Objekt.

  Der `JobCommand`, der diesen Auftrag ausführt.

- `DefaultArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die Standardargumente für jede Ausführung dieses Auftrags, angegeben als Name-Wert-Paare.

  Sie können hier Argumente angeben, die Ihr eigenes Jobausführungsskript verwendet, sowie Argumente, die AWS Glue selbst verwendet werden.

  Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

  Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Spark-Aufträgen angeben können, finden Sie im Thema Spezielle Parameter, die von AWS Glue verwendet werden im Entwicklerhandbuch.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Ray-Aufträgen angeben können, finden Sie unter Verwenden von Auftragsparametern in Ray-Aufträgen im Entwicklerhandbuch.

- `NonOverridableArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Argumente für diesen Auftrag, die beim Bereitstellen von Auftragsargumenten in einer Auftragsausführung nicht überschrieben werden und als Name-Wert-Paare angegeben werden.

- `Connections` – Ein ConnectionsList-Objekt.

  Die Verbindungen, die für diesen Auftrag verwendet werden.

- `MaxRetries` – Zahl (Ganzzahl).

  Gibt an, wie oft dieser Job nach einem JobRun Fehlschlag maximal wiederholt werden soll.

- `AllocatedCapacity` – Zahl (Ganzzahl).

  Diese Feld ist als veraltet gekennzeichnet. Verwenden Sie stattdessen `MaxCapacity`.

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die den Ausführungen dieses Jobs zugewiesen sind. Sie können mindestens 2 zuweisen DPUs; die Standardeinstellung ist 10. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Die Auftrag-Zeitüberschreitung in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den `TIMEOUT`-Status wechselt.

  Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

  Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

  Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

  Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `MaxCapacity` – Nummer (doppelt).

  Für Glue-Jobs der Version 1.0 oder früher, die den Standard-Worker-Typ verwenden, die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher besteht. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Für Aufträge der Glue-Version 2.0 oder höher können Sie keine `Maximum capacity` angeben. Stattdessen sollten Sie einen `Worker type` und die `Number of workers` festlegen.

Setzen Sie nicht `MaxCapacity`, wenn Sie `WorkerType` und `NumberOfWorkers` verwenden.

Der Wert, der `MaxCapacity` zugewiesen werden kann, ist davon abhängig, ob Sie eine Python-Shell-Aufgabe, eine Apache-Spark-ETL-Aufgabe oder eine Apache-Spark-Streaming-ETL-Aufgabe ausführen:

- Wenn Sie einen Python-Shell-Auftrag (`JobCommand.Name`="pythonshell") angeben, können Sie entweder 0,0625 oder 1 DPU zuweisen. Der Standardwert ist 0,0625 DPU.
- Wenn Sie einen Apache Spark-ETL-Job (`JobCommand.Name`="glueetl“) oder einen Apache Spark-Streaming-ETL-Job (`JobCommand.Name`="gluestreaming“) angeben, können Sie zwischen 2 und 100 zuweisen. DPUs Die Standardeinstellung ist 10. DPUs Diesem Auftragstyp dürfen keine DPU-Bruchteile zugeteilt werden.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird.

  AWS Glue bietet mehrere Workertypen, um unterschiedlichen Workload-Anforderungen gerecht zu werden:

  G-Worker-Typen (Computer-Worker für allgemeine Zwecke):

  - G.1X: 1 DPU (4 VCPUs, 16 GB Arbeitsspeicher, 94 GB Festplatte)
  - G.2X: 2 DPU (8 VCPUs, 32 GB Speicher, 138 GB Festplatte)
  - G.4X: 4 DPU (16 VCPUs, 64 GB Speicher, 256 GB Festplatte)
  - G.8X: 8 DPU (32 VCPUs, 128 GB Speicher, 512 GB Festplatte)
  - G.12X: 12 DPU (48 VCPUs, 192 GB Speicher, 768 GB Festplatte)
  - G.16X: 16 DPU (64 VCPUs, 256 GB Speicher, 1024 GB Festplatte)

  R-Worker-Typen (speicheroptimierte Worker):

  - R.1X: 1 M-DPU (4 VCPUs, 32 GB Speicher)
  - R.2X: 2 M-DPU (8 VCPUs, 64 GB Speicher)
  - R.4X: 4 M-DPU (16 VCPUs, 128 GB Speicher)
  - R.8X: 8 M-DPU (32 VCPUs, 256 GB Speicher)

- `NumberOfWorkers` – Zahl (Ganzzahl).

Die Anzahl der Worker eines definierten `workerType` die zugewiesenj werden, wenn ein Auftrag ausgeführt wird.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem Auftrag verwendet werden soll.

- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsbenachrichtigung an.

- `Running` – Boolesch.

  Dieses Feld ist für zukünftige Zwecke reserviert.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  `GlueVersion`Ermittelt in Spark-Jobs die Versionen von Apache Spark und Python, die in einem Job AWS Glue verfügbar sind. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird.

  In Ray-Aufträge sollte die `GlueVersion` auf `4.0` oder höher eingestellt sein. Welche Versionen von Ray, Python und zusätzlichen Bibliotheken in Ihrem Ray-Auftrag verfügbar sind, wird jedoch durch die `Runtime`-Parameter des Auftragsbefehls bestimmt.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

  Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

- `CodeGenConfigurationNodes` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Jeder Wert ist ein A CodeGenConfigurationNode-Objekt.

  Die Darstellung eines gerichteten azyklischen Grafiken, auf dem sowohl die visuelle Komponente von Glue Studio als auch die Codegenerierung von Glue Studio basieren.

- `ExecutionClass` – UTF-8-Zeichenfolge, nicht länger als 16 Bytes (gültige Werte: `FLEX=""` | `STANDARD=""`).

Gibt an, ob der Auftrag mit einer Standard- oder einer flexiblen Ausführungsklasse ausgeführt wird. Die Standardausführungsklasse ist optimal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

Die flexible Ausführungsklasse ist geeignet für zeitunabhängige Aufträge, deren Start- und Abschlusszeiten variieren können.

Nur Jobs mit AWS Glue Version 3.0 und höher und dem Befehlstyp dürfen `ExecutionClass` auf gesetzt `glueetl` werden`FLEX`. Die flexible Ausführungsklasse ist für Spark-Aufträge verfügbar.

- `SourceControlDetails` – Ein SourceControlDetails-Objekt.

  Die Details für eine Quellcodeverwaltungskonfiguration für einen Auftrag, die die Synchronisation von Auftragsartefakten zu oder von einem Remote-Repository ermöglichen.

- `MaintenanceWindow` – UTF-8-Zeichenfolge, die Custom string pattern #34 entspricht.

  Dieses Feld gibt einen Wochentag und eine Stunde für ein Wartungsfenster für Streaming-Jobs an. AWS Glue führt regelmäßig Wartungsarbeiten durch. Während dieser Wartungsfenster müssen AWS Glue Sie Ihre Streaming-Jobs neu starten.

  AWS Glue wird den Job innerhalb von 3 Stunden nach Ablauf des angegebenen Wartungsfensters neu starten. Wenn Sie beispielsweise das Wartungsfenster für Montag um 10:00 Uhr GMT einrichten, werden Ihre Jobs zwischen 10:00 Uhr GMT und 13:00 Uhr GMT neu gestartet.

- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines mit dem Job verknüpften AWS Glue Nutzungsprofils.

ExecutionProperty Struktur

Eine Ausführungseigenschaft eines Auftrags.

Felder

- `MaxConcurrentRuns` – Zahl (Ganzzahl).

  Die maximale Anzahl der gleichzeitigen Ausführungen, die für diesen Auftrag zulässig sind. Der Standardwert ist 1. Bei Erreichen dieser Schwelle wird ein Fehler zurückgegeben. Der Höchstwert, den Sie angeben können, wird durch ein Service Limit gesteuert.

NotificationProperty Struktur

Gibt die Konfigurationseigenschaften einer Benachrichtigung an.

Felder

- `NotifyDelayAfter` – Zahl (ganze Zahl), mindestens 1.

  Nach dem Start eines Auftragslaufs gibt dies die Anzahl der Minuten an, die gewartet werden muss, bevor eine Benachrichtigung über die Verzögerung eines Auftragslaufs gesendet wird.

JobCommand Struktur

Gibt den Code an, der während einer Auftragsausführung ausgeführt wird.

Felder

- `Name` – UTF-8-Zeichenfolge.

  Der Name des Auftragsbefehls. Für einen Apache-Spark-ETL-Auftrag muss dies `glueetl` sein. Bei einem Python-Shell-Auftrag muss dies `pythonshell` sein. Für eine Apache-Spark-Streaming-ETL-Aufgabe muss dies sein `gluestreaming`. Für einen Ray-Auftrag muss dies `glueray` sein.
- `ScriptLocation` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Gibt den Amazon Simple Storage Service (Amazon S3)-Pfad zu einem Skript zum Ausführen eines Auftrags an.
- `PythonVersion` – UTF-8-Zeichenfolge, die Custom string pattern #48 entspricht.

  Die zum Ausführen eines Python-Shell-Auftrags verwendete Python-Version. Zulässige Werte sind 2 und 3.
- `Runtime` – UTF-8-Zeichenfolge, nicht mehr als 64 Bytes lang, passend zum Custom string pattern #33.

  In Ray-Aufträgen wird Runtime verwendet, um die in Ihrer Umgebung verfügbaren Versionen von Ray, Python und zusätzlichen Bibliotheken anzugeben. Dieses Feld wird in anderen Auftragsarten nicht verwendet. Die Werte der unterstützten Laufzeitumgebung finden Sie unter Unterstützte Ray-Laufzeitumgebungen im AWS Glue Entwicklerhandbuch.

ConnectionsList Struktur

Gibt die Verbindungen an, die von einem Auftrag verwendet werden.

Felder

- `Connections`— Ein Array von UTF-8-Strings, nicht mehr als 1000 Zeichenketten.

  Eine Liste der Verbindungen, die vom Auftrag verwendet werden.

JobUpdate Struktur

Gibt Informationen zum Aktualisieren einer vorhandenen Auftragsdefinition an. Die vorherige Auftragsdefinition wird von diesen Informationen vollständig überschrieben.

Felder

- `JobMode` – UTF-8-Zeichenfolge (zulässige Werte: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Ein Modus, der beschreibt, wie ein Job erstellt wurde. Gültige Werte für sind:

  - `SCRIPT`- Der Job wurde mit dem AWS Glue Studio-Skripteditor erstellt.
  - `VISUAL`- Der Job wurde mit dem Visual Editor von AWS Glue Studio erstellt.
  - `NOTEBOOK`- Der Job wurde mit einem interaktiven Sitzungsnotizbuch erstellt.

  Wenn das `JobMode` Feld fehlt oder Null `SCRIPT` ist, wird es als Standardwert zugewiesen.
- `JobRunQueuingEnabled` – Boolesch.

  Gibt an, ob die Warteschlange für die Auftragsausführung für diesen Job aktiviert ist.

  Der Wert true bedeutet, dass das Job Run Queuing für die Jobausführungen aktiviert ist. Wenn der Wert falsch oder nicht ausgefüllt ist, werden die Auftragsausführungen nicht in die Warteschlange aufgenommen.

  Wenn dieses Feld nicht mit dem Wert übereinstimmt, der in der Auftragsausführung festgelegt wurde, wird der Wert aus dem Feld für die Auftragsausführung verwendet.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibung des Auftrags, der definiert wird.

- `LogUri` – UTF-8-Zeichenfolge.

  Dieses Feld ist für zukünftige Zwecke reserviert.

- `Role` – UTF-8-Zeichenfolge.

  Der Name oder Amazon-Ressourcenname (ARN) der IAM-Rolle, die diesem Auftrag zugeordnet ist (erforderlich).

- `ExecutionProperty` – Ein ExecutionProperty-Objekt.

  Eine `ExecutionProperty`, die die maximale Anzahl der gleichzeitigen Ausführungen angibt, die für diesen Auftrag zulässig sind.

- `Command` – Ein JobCommand-Objekt.

  Der `JobCommand`, der diesen Auftrag ausführt (erforderlich).

- `DefaultArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die Standardargumente für jede Ausführung dieses Auftrags, angegeben als Name-Wert-Paare.

  Sie können hier Argumente angeben, die Ihr eigenes Jobausführungsskript verwendet, sowie Argumente, die AWS Glue selbst verwendet werden.

  Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

  Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Spark-Aufträgen angeben können, finden Sie im Thema Spezielle Parameter, die von AWS Glue verwendet werden im Entwicklerhandbuch.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Ray-Aufträgen angeben können, finden Sie unter Verwenden von Auftragsparametern in Ray-Aufträgen im Entwicklerhandbuch.

- `NonOverridableArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Argumente für diesen Auftrag, die beim Bereitstellen von Auftragsargumenten in einer Auftragsausführung nicht überschrieben werden und als Name-Wert-Paare angegeben werden.

- `Connections` – Ein ConnectionsList-Objekt.

  Die Verbindungen, die für diesen Auftrag verwendet werden.

- `MaxRetries` – Zahl (Ganzzahl).

  Die maximale Anzahl der Wiederholungsversuche für diesen Auftrag, wenn er fehlschlägt.

- `AllocatedCapacity` – Zahl (Ganzzahl).

  Diese Feld ist als veraltet gekennzeichnet. Verwenden Sie stattdessen `MaxCapacity`.

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem Job zugewiesen werden sollen. Sie können mindestens 2 zuweisen DPUs; die Standardeinstellung ist 10. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Die Auftrag-Zeitüberschreitung in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den `TIMEOUT`-Status wechselt.

  Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

  Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

  Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

  Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `MaxCapacity` – Nummer (doppelt).

  Für Glue-Jobs der Version 1.0 oder früher, die den Standard-Worker-Typ verwenden, die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher besteht. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Bei Aufträgen ab Glue-Version 2.0 können Sie keine `Maximum capacity` angeben. Stattdessen sollten Sie einen `Worker type` und die `Number of workers` festlegen.

  Setzen Sie nicht `MaxCapacity`, wenn Sie `WorkerType` und `NumberOfWorkers` verwenden.

  Der Wert, der `MaxCapacity` zugewiesen werden kann, ist davon abhängig, ob Sie eine Python-Shell-Aufgabe, eine Apache-Spark-ETL-Aufgabe oder eine Apache-Spark-Streaming-ETL-Aufgabe ausführen:

  - Wenn Sie einen Python-Shell-Auftrag (`JobCommand.Name`="pythonshell") angeben, können Sie entweder 0,0625 oder 1 DPU zuweisen. Der Standardwert ist 0,0625 DPU.
  - Wenn Sie einen Apache Spark-ETL-Job (`JobCommand.Name`="glueetl“) oder einen Apache Spark-Streaming-ETL-Job (`JobCommand.Name`="gluestreaming“) angeben, können Sie zwischen 2 und 100 zuweisen. DPUs Die Standardeinstellung ist 10. DPUs Diesem Auftragstyp dürfen keine DPU-Bruchteile zugeteilt werden.
- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird. Akzeptiert einen Wert von G.1X, G.2X, G.4X, G.8X oder G.025X für Spark-Aufträge. Akzeptiert den Wert Z.2X für Ray-Aufträge. Weitere Informationen finden Sie unter Definieren von Jobeigenschaften für Spark-Jobs
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die zugewiesenj werden, wenn ein Auftrag ausgeführt wird.
- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem Auftrag verwendet werden soll.

- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsbenachrichtigung an.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  `GlueVersion`Ermittelt in Spark-Jobs die Versionen von Apache Spark und Python, die in einem Job AWS Glue verfügbar sind. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird.

  In Ray-Aufträge sollte die `GlueVersion` auf `4.0` oder höher eingestellt sein. Welche Versionen von Ray, Python und zusätzlichen Bibliotheken in Ihrem Ray-Auftrag verfügbar sind, wird jedoch durch die `Runtime`-Parameter des Auftragsbefehls bestimmt.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

  Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

- `CodeGenConfigurationNodes` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Jeder Wert ist ein A CodeGenConfigurationNode-Objekt.

  Die Darstellung eines gerichteten azyklischen Grafiken, auf dem sowohl die visuelle Komponente von Glue Studio als auch die Codegenerierung von Glue Studio basieren.

- `ExecutionClass` – UTF-8-Zeichenfolge, nicht länger als 16 Bytes (gültige Werte: `FLEX=""` | `STANDARD=""`).

  Gibt an, ob der Auftrag mit einer Standard- oder einer flexiblen Ausführungsklasse ausgeführt wird. Die Standardausführungsklasse ist ideal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

  Die flexible Ausführungsklasse ist geeignet für zeitunabhängige Aufträge, deren Start- und Abschlusszeiten variieren können.

  Nur Jobs mit AWS Glue Version 3.0 und höher und dem Befehlstyp dürfen `ExecutionClass` auf gesetzt `glueetl` werden`FLEX`. Die flexible Ausführungsklasse ist für Spark-Aufträge verfügbar.

- `SourceControlDetails` – Ein SourceControlDetails-Objekt.

Die Details für eine Quellcodeverwaltungskonfiguration für einen Auftrag, die die Synchronisation von Auftragsartefakten zu oder von einem Remote-Repository ermöglichen.

- `MaintenanceWindow` – UTF-8-Zeichenfolge, die Custom string pattern #34 entspricht.

  Dieses Feld gibt einen Wochentag und eine Stunde für ein Wartungsfenster für Streaming-Jobs an. AWS Glue führt regelmäßig Wartungsarbeiten durch. Während dieser Wartungsfenster müssen AWS Glue Sie Ihre Streaming-Jobs neu starten.

  AWS Glue wird den Job innerhalb von 3 Stunden nach Ablauf des angegebenen Wartungsfensters neu starten. Wenn Sie beispielsweise das Wartungsfenster für Montag um 10:00 Uhr GMT einrichten, werden Ihre Jobs zwischen 10:00 Uhr GMT und 13:00 Uhr GMT neu gestartet.
- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines mit dem Job verknüpften AWS Glue Nutzungsprofils.

SourceControlDetails Struktur

Die Details für eine Quellcodeverwaltungskonfiguration für einen Auftrag, die die Synchronisation von Auftragsartefakten zu oder von einem Remote-Repository ermöglichen.

Felder

- `Provider` – UTF-8-Zeichenfolge (zulässige Werte: `GITHUB` | `AWS_CODE_COMMIT`).

  Der Anbieter für das Remote-Repository.
- `Repository` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Der Name des Remote-Repositorys, das die Auftragsartefakte enthält.
- `Owner` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Der Besitzer des Remote-Repositorys, das die Auftragsartefakte enthält.
- `Branch` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Eine optionale Verzweigung im Remote-Repository.
- `Folder` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Ein optionaler Ordner im Remote-Repository.

- `LastCommitId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Die letzte Commit-ID für ein Commit im Remote-Repository.

- `LastSyncTimestamp` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Das Datum und die Uhrzeit, an denen die letzte Auftragssynchronisierung durchgeführt wurde.

- `AuthStrategy` – UTF-8-Zeichenfolge (zulässige Werte: `PERSONAL_ACCESS_TOKEN` | `AWS_SECRETS_MANAGER`).

  Die Art der Authentifizierung, bei der es sich um ein in AWS Secrets Manager gespeichertes Authentifizierungstoken oder um ein persönliches Zugriffstoken handeln kann.

- `AuthToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Der Wert eines Autorisierungstokens.

Operationen

- CreateJob Aktion (Python: create_job)
- UpdateJob Aktion (Python: update_job)
- GetJob Aktion (Python: get_job)
- GetJobs Aktion (Python: get_jobs)
- DeleteJob Aktion (Python: delete_job)
- ListJobs Aktion (Python: list_jobs)
- BatchGetJobs Aktion (Python: batch_get_jobs)

CreateJob Aktion (Python: create_job)

Erstellt eine neue Auftragsdefinition

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name, den Sie dieser Auftragsdefinition zuweisen. Er muss in Ihrem -Konto eindeutig sein.

- `JobMode` – UTF-8-Zeichenfolge (zulässige Werte: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

Ein Modus, der beschreibt, wie ein Job erstellt wurde. Gültige Werte für sind:

- `SCRIPT`- Der Job wurde mit dem AWS Glue Studio-Skripteditor erstellt.
- `VISUAL`- Der Job wurde mit dem Visual Editor von AWS Glue Studio erstellt.
- `NOTEBOOK`- Der Job wurde mit einem interaktiven Sitzungsnotizbuch erstellt.

Wenn das `JobMode` Feld fehlt oder Null `SCRIPT` ist, wird es als Standardwert zugewiesen.

- `JobRunQueuingEnabled` – Boolesch.

  Gibt an, ob die Warteschlange für die Auftragsausführung für diesen Job aktiviert ist.

  Der Wert true bedeutet, dass das Job Run Queuing für die Jobausführungen aktiviert ist. Wenn der Wert falsch ist oder nicht ausgefüllt wird, werden die Auftragsausführungen nicht in die Warteschlange aufgenommen.

  Wenn dieses Feld nicht mit dem Wert übereinstimmt, der in der Auftragsausführung festgelegt wurde, wird der Wert aus dem Feld für die Auftragsausführung verwendet.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Beschreibung des Auftrags, der definiert wird.

- `LogUri` – UTF-8-Zeichenfolge.

  Dieses Feld ist für zukünftige Zwecke reserviert.

- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name oder der Amazon-Ressourcenname (ARN) der IAM-Rolle, die diesem Auftrag zugeordnet ist.

- `ExecutionProperty` – Ein ExecutionProperty-Objekt.

  Eine `ExecutionProperty`, die die maximale Anzahl der gleichzeitigen Ausführungen angibt, die für diesen Auftrag zulässig sind.

- `Command` – Erforderlich: Ein JobCommand-Objekt.

  Der `JobCommand`, der diesen Auftrag ausführt.

- `DefaultArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

Jeder Wert ist eine UTF-8-Zeichenfolge.

Die Standardargumente für jede Ausführung dieses Auftrags, angegeben als Name-Wert-Paare.

Sie können hier Argumente angeben, die Ihr eigenes Jobausführungsskript verwendet, sowie Argumente, die AWS Glue selbst verwendet werden.

Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Spark-Aufträgen angeben können, finden Sie im Thema Spezielle Parameter, die von AWS Glue verwendet werden im Entwicklerhandbuch.

Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Ray-Aufträgen angeben können, finden Sie unter Verwenden von Auftragsparametern in Ray-Aufträgen im Entwicklerhandbuch.

- `NonOverridableArguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Argumente für diesen Auftrag, die beim Bereitstellen von Auftragsargumenten in einer Auftragsausführung nicht überschrieben werden und als Name-Wert-Paare angegeben werden.

- `Connections` – Ein ConnectionsList-Objekt.

  Die Verbindungen, die für diesen Auftrag verwendet werden.

- `MaxRetries` – Zahl (Ganzzahl).

  Die maximale Anzahl der Wiederholungsversuche für diesen Auftrag, wenn er fehlschlägt.

- `AllocatedCapacity` – Zahl (Ganzzahl).

  Dieser Parameter ist veraltet. Verwenden Sie stattdessen `MaxCapacity`.

Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem Job zugewiesen werden sollen. Sie können mindestens 2 zuweisen DPUs; die Standardeinstellung ist 10. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Die Auftrag-Zeitüberschreitung in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den `TIMEOUT`-Status wechselt.

  Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

  Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

  Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

  Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `MaxCapacity` – Nummer (doppelt).

  Für Glue-Jobs der Version 1.0 oder früher, die den Standard-Worker-Typ verwenden, die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher besteht. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Bei Aufträgen ab Glue-Version 2.0 können Sie keine `Maximum capacity` angeben. Stattdessen sollten Sie einen `Worker type` und die `Number of workers` festlegen.

  Setzen Sie nicht `MaxCapacity`, wenn Sie `WorkerType` und `NumberOfWorkers` verwenden.

  Der Wert, der `MaxCapacity` zugewiesen werden kann, ist davon abhängig, ob Sie eine Python-Shell-Aufgabe, eine Apache-Spark-ETL-Aufgabe oder eine Apache-Spark-Streaming-ETL-Aufgabe ausführen:

  - Wenn Sie einen Python-Shell-Auftrag (`JobCommand.Name`="pythonshell") angeben, können Sie entweder 0,0625 oder 1 DPU zuweisen. Der Standardwert ist 0,0625 DPU.
  - Wenn Sie einen Apache Spark-ETL-Job (`JobCommand.Name`="glueetl“) oder einen Apache Spark-Streaming-ETL-Job (`JobCommand.Name`="gluestreaming“) angeben, können Sie zwischen 2 und 100 zuweisen. DPUs Die Standardeinstellung ist 10. DPUs Diesem Auftragstyp dürfen keine DPU-Bruchteile zugeteilt werden.
- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem Auftrag verwendet werden soll.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die mit diesem Auftrag zu verwendenden Tags. Sie können Tags verwenden, um den Zugriff auf den Auftrag einzuschränken. Weitere Informationen zu Tags in AWS Glue finden Sie unter AWS Tags in AWS Glue im Entwicklerhandbuch.
- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsbenachrichtigung an.
- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  `GlueVersion`Ermittelt in Spark-Jobs die Versionen von Apache Spark und Python, die in einem Job AWS Glue verfügbar sind. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird.

  In Ray-Aufträge sollte die `GlueVersion` auf `4.0` oder höher eingestellt sein. Welche Versionen von Ray, Python und zusätzlichen Bibliotheken in Ihrem Ray-Auftrag verfügbar sind, wird jedoch durch die `Runtime`-Parameter des Auftragsbefehls bestimmt.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

  Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die zugewiesenj werden, wenn ein Auftrag ausgeführt wird.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird. Akzeptiert einen Wert von G.1X, G.2X, G.4X, G.8X oder G.025X für Spark-Aufträge. Akzeptiert den Wert Z.2X für Ray-Aufträge.

  - Beim `G.1X` Worker-Typ wird jeder Worker einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 94 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.2X` Worker-Typ wird jeder Worker 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 138 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.4X` Worker-Typ wird jeder Worker 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Workertyp ist nur für Spark ETL-Jobs der AWS Glue Version 3.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland) und Europa (Stockholm).
  - Für den `G.8X` Worker-Typ wird jeder Worker 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Jobs der AWS Glue Version 3.0 oder höher in denselben AWS Regionen verfügbar, die auch für den Worker-Typ unterstützt werden. `G.4X`
  - Für den `G.025X` Worker-Typ entspricht jeder Worker 0,25 DPU (2 VCPUs, 4 GB Arbeitsspeicher) mit 84 GB Festplatte und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Streaming-Aufträge mit geringem Volumen. Dieser Worker-Typ ist nur für Streaming-Jobs der AWS Glue Version 3.0 oder höher verfügbar.

  - Für den `Z.2X` Worker-Typ wird jeder Worker 2 M-DPU (8 VCPUs, 64 GB Arbeitsspeicher) mit 128 GB Festplatte zugeordnet und bietet auf Basis des Autoscalers bis zu 8 Ray-Worker.
- `CodeGenConfigurationNodes` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die Custom string pattern #58 entspricht.

  Jeder Wert ist ein A CodeGenConfigurationNode-Objekt.

  Die Darstellung eines gerichteten azyklischen Grafiken, auf dem sowohl die visuelle Komponente von Glue Studio als auch die Codegenerierung von Glue Studio basieren.
- `ExecutionClass` – UTF-8-Zeichenfolge, nicht länger als 16 Bytes (gültige Werte: `FLEX=""` | `STANDARD=""`).

  Gibt an, ob der Auftrag mit einer Standard- oder einer flexiblen Ausführungsklasse ausgeführt wird. Die Standardausführungsklasse ist ideal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

  Die flexible Ausführungsklasse ist geeignet für zeitunabhängige Aufträge, deren Start- und Abschlusszeiten variieren können.

  Nur Jobs mit AWS Glue Version 3.0 und höher und mit dem Befehlstyp dürfen auf `glueetl` gesetzt werden. `ExecutionClass` `FLEX` Die flexible Ausführungsklasse ist für Spark-Aufträge verfügbar.
- `SourceControlDetails` – Ein SourceControlDetails-Objekt.

  Die Details für eine Quellcodeverwaltungskonfiguration für einen Auftrag, die die Synchronisation von Auftragsartefakten zu oder von einem Remote-Repository ermöglichen.
- `MaintenanceWindow` – UTF-8-Zeichenfolge, die Custom string pattern #34 entspricht.

  Dieses Feld gibt einen Wochentag und eine Stunde für ein Wartungsfenster für Streaming-Jobs an. AWS Glue führt regelmäßig Wartungsarbeiten durch. Während dieser Wartungsfenster müssen AWS Glue Sie Ihre Streaming-Jobs neu starten.

  AWS Glue wird den Job innerhalb von 3 Stunden nach Ablauf des angegebenen Wartungsfensters neu starten. Wenn Sie beispielsweise das Wartungsfenster für Montag um 10:00 Uhr GMT einrichten, werden Ihre Jobs zwischen 10:00 Uhr GMT und 13:00 Uhr GMT neu gestartet.
- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name eines mit dem Job verknüpften AWS Glue Nutzungsprofils.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der eindeutige Name, der für diese Auftragsdefinition angegeben wurde.

Fehler

- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `AlreadyExistsException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateJob Aktion (Python: update_job)

Aktualisiert eine vorhandene Auftragsdefinition. Die vorherige Auftragsdefinition wird von diesen Informationen vollständig überschrieben.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, die aktualisiert werden soll.
- `JobUpdate` – Erforderlich: Ein JobUpdate-Objekt.

  Gibt die Werte zum Aktualisieren der Auftragsdefinition an. Eine nicht angegebene Konfiguration wird entfernt oder auf Standardwerte zurückgesetzt.
- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name eines mit dem Job verknüpften AWS Glue Nutzungsprofils.

Antwort

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen der aktualisierten Auftragsdefinition an.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

GetJob Aktion (Python: get_job)

Ruft eine vorhandene Auftragsdefinition ab.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der abzurufenden Auftragsdefinition.

Antwort

- `Job` – Ein Aufgabe-Objekt.

  Die angeforderte Auftragsdefinition.

Fehler

- `InvalidInputException`

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobs Aktion (Python: get_jobs)

Ruft alle aktuellen Auftragsdefinitionen ab.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der Antwort.

Antwort

- `Jobs` – Ein Array mit Aufgabe-Objekten.

  Eine Liste der Auftragsdefinitionen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungstoken, falls noch nicht alle Auftragsdefinitionen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

DeleteJob Aktion (Python: delete_job)

Löscht eine angegebene Auftragsdefinition. Wenn die Auftragsdefinition nicht gefunden wird, wird keine Ausnahme ausgelöst.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu löschenden Auftragsdefinition.

Antwort

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, die gelöscht wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

ListJobs Aktion (Python: list_jobs)

Ruft die Namen aller Jobressourcen in diesem AWS Konto oder der Ressourcen mit dem angegebenen Tag ab. Mit dieser Operation können Sie sehen, welche Ressourcen in Ihrem Konto verfügbar sind, sowie deren Namen.

Diese Operation akzeptiert das optionale `Tags`-Feld, das Sie als Filter für die Antwort verwenden können, so dass markierte Ressourcen als Gruppe abgerufen werden können. Wenn Sie die Tag-Filterung verwenden, werden nur Ressourcen mit dem Tag abgerufen.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Gibt an, das nur diese markierten Ressourcen zurückgegeben werden sollen.

Antwort

- `JobNames` – Ein UTF-8-Zeichenfolgen-Array.

  Die Namen aller Aufträge im Konto oder der Aufträge mit den angegebenen Tags.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetJobs Aktion (Python: batch_get_jobs)

Gibt eine Liste von Ressourcen-Metadaten für eine bestimmte Liste von Auftragsnamen zurück. Nach dem Aufrufen der `ListJobs`-Operation können Sie diese Operation aufrufen, um auf die Daten zuzugreifen, für die Ihnen Berechtigungen erteilt wurden. Dieser Vorgang unterstützt alle IAM-Berechtigungen, einschließlich Berechtigungsbedingungen, die Tags verwenden.

Anforderung

- `JobNames` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste von Auftragsnamen, die von der `ListJobs`-Operation als Namen zurückgegeben werden können.

Antwort

- `Jobs` – Ein Array mit Aufgabe-Objekten.

  Eine Liste der Auftragsdefinitionen.
- `JobsNotFound` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Namen nicht gefundener Aufträgen.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

Auftragsausführungen

Die Jobs Runs-API beschreibt die Datentypen und die API im Zusammenhang mit dem Starten, Stoppen oder Anzeigen von Auftragsausführungen und dem Zurücksetzen von Job-Lesezeichen unter. AWS Glue Der Verlauf der Auftragsausführung ist 90 Tage lang für Ihren Workflow und Ihre Auftragsausführung zugänglich.

Datentypen

- JobRun Struktur
- Vorgängerstruktur
- JobBookmarkEntry Struktur
- BatchStopJobRunSuccessfulSubmission Struktur
- BatchStopJobRunError Struktur
- NotificationProperty Struktur

JobRun Struktur

Enthält Informationen zu einer Auftragsausführung.

Felder

- `Id` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID dieser Auftragsausführung.
- `Attempt` – Zahl (Ganzzahl).

  Die Anzahl der Versuche für die Ausführung dieses Auftrags.
- `PreviousRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der vorherigen Ausführung dieses Auftrags. Beispiel: Die `JobRunId` in der `StartJobRun`-Aktion.
- `TriggerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der diese Auftragsausführung gestartet hat.
- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, die in dieser Ausführung verwendet wird.
- `JobMode` – UTF-8-Zeichenfolge (zulässige Werte: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Ein Modus, der beschreibt, wie ein Job erstellt wurde. Gültige Werte für sind:
  - `SCRIPT`- Der Job wurde mit dem AWS Glue Studio-Skripteditor erstellt.
  - `VISUAL`- Der Job wurde mit dem Visual Editor von AWS Glue Studio erstellt.
  - `NOTEBOOK`- Der Job wurde mit einem interaktiven Sitzungsnotizbuch erstellt.

  Wenn das `JobMode` Feld fehlt oder Null `SCRIPT` ist, wird es als Standardwert zugewiesen.
- `JobRunQueuingEnabled` – Boolesch.

  Gibt an, ob die Warteschlange für die Auftragsausführung aktiviert ist.

  Der Wert true bedeutet, dass die Warteschlange für die Auftragsausführung aktiviert ist. Wenn der Wert falsch oder nicht ausgefüllt ist, wird der ausgeführte Job nicht in die Warteschlange aufgenommen.
- `StartedOn` – Zeitstempel.

Das Datum und die Uhrzeit, an denen diese Auftragsausführung gestartet wurde

- `LastModifiedOn` – Zeitstempel.

  Der letzte Zeitpunkt, an dem diese Auftragsausführung geändert wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, an denen diese Auftragsausführung abgeschlossen wurde

- `JobRunState`— UTF-8-Zeichenfolge (gültige Werte: `STARTING` | | | `RUNNING` | | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` |`WAITING`). `EXPIRED`

  Den aktuellen Status der Auftragsausführung. Weitere Informationen über den Status von Aufgaben, die in ungewöhnlicher Weise beendet wurden, finden Sie unter AWS Glue - Aufgabenausführungsstatus.

- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die mit dieser Ausführung verknüpften Auftragsargumente. Für diese Auftragsauführung ersetzen sie den Satz der Standardargumente direkt in der Auftragsdefinition.

  Sie können hier Argumente angeben, die Ihr eigenes Job-Ausführungsskript verwendet, sowie Argumente, die selbst verwendet werden. AWS Glue

  Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

  Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Spark-Aufträgen angeben können, finden Sie im Thema Spezielle Parameter, die von AWS Glue verwendet werden im Entwicklerhandbuch.

Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Ray-Aufträgen angeben können, finden Sie unter Verwenden von Auftragsparametern in Ray-Aufträgen im Entwicklerhandbuch.

- `ErrorMessage` – UTF-8-Zeichenfolge.

  Eine Fehlermeldung, die mit dieser Auftragsausführung verknüpft ist.

- `PredecessorRuns` – Ein Array mit Vorgänger-Objekten.

  Eine Liste der Vorgänger zu dieser Auftragsausführung.

- `AllocatedCapacity` – Zahl (Ganzzahl).

  Diese Feld ist als veraltet gekennzeichnet. Verwenden Sie stattdessen `MaxCapacity`.

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem zugewiesen sind JobRun. Es DPUs können 2 bis 100 zugewiesen werden; die Standardeinstellung ist 10. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Auftragsausführung Ressourcen verbraucht wurden.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Der`JobRun`-Timeoutwert in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den `TIMEOUT`-Status wechselt. Dieser Wert überschreibt den Wert der Zeitüberschreitung im übergeordneten Auftrag.

  Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

  Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

  Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

  Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `MaxCapacity` – Nummer (doppelt).

  Für Glue-Jobs der Version 1.0 oder früher, die den Standard-Worker-Typ verwenden, die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Bei Aufträgen ab Glue-Version 2.0 können Sie keine `Maximum capacity` angeben. Stattdessen sollten Sie einen `Worker type` und die `Number of workers` festlegen.

  Setzen Sie nicht `MaxCapacity`, wenn Sie `WorkerType` und `NumberOfWorkers` verwenden.

  Der Wert, der `MaxCapacity` zugewiesen werden kann, ist davon abhängig, ob Sie eine Python-Shell-Aufgabe, eine Apache-Spark-ETL-Aufgabe oder eine Apache-Spark-Streaming-ETL-Aufgabe ausführen:

  - Wenn Sie einen Python-Shell-Auftrag (`JobCommand.Name`="pythonshell") angeben, können Sie entweder 0,0625 oder 1 DPU zuweisen. Der Standardwert ist 0,0625 DPU.
  - Wenn Sie einen Apache Spark-ETL-Job (`JobCommand.Name`="glueetl“) oder einen Apache Spark-Streaming-ETL-Job (`JobCommand.Name`="gluestreaming“) angeben, können Sie zwischen 2 und 100 zuweisen. DPUs Die Standardeinstellung ist 10. DPUs Diesem Auftragstyp dürfen keine DPU-Bruchteile zugeteilt werden.
- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird. Akzeptiert einen Wert von G.1X, G.2X, G.4X, G.8X oder G.025X für Spark-Aufträge. Akzeptiert den Wert Z.2X für Ray-Aufträge.

  - Beim `G.1X` Worker-Typ wird jeder Worker einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 94 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.2X` Worker-Typ wird jeder Worker 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 138 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.

  - Für den `G.4X` Worker-Typ wird jeder Worker 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Workertyp ist nur für Spark ETL-Jobs der AWS Glue Version 3.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland) und Europa (Stockholm).
  - Für den `G.8X` Worker-Typ wird jeder Worker 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Jobs der AWS Glue Version 3.0 oder höher in denselben AWS Regionen verfügbar, die auch für den Worker-Typ unterstützt werden. `G.4X`
  - Für den `G.025X` Worker-Typ entspricht jeder Worker 0,25 DPU (2 VCPUs, 4 GB Arbeitsspeicher) mit 84 GB Festplatte und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Streaming-Aufträge mit geringem Volumen. Dieser Worker-Typ ist nur für Streaming-Jobs der AWS Glue Version 3.0 oder höher verfügbar.
  - Für den `Z.2X` Worker-Typ wird jeder Worker 2 M-DPU (8 VCPUs, 64 GB Arbeitsspeicher) mit 128 GB Festplatte zugeordnet und bietet auf Basis des Autoscalers bis zu 8 Ray-Worker.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die zugewiesenj werden, wenn ein Auftrag ausgeführt wird.
- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem Auftrag verwendet werden soll.
- `LogGroupName` – UTF-8-Zeichenfolge.

  Der Name der Protokollgruppe für die sichere Protokollierung, die in Amazon CloudWatch serverseitig verschlüsselt werden kann. AWS KMS Dieser Name kann `/aws-glue/jobs/` sein, in diesem Fall ist die Standard-Verschlüsselung `NONE`. Wenn Sie einen Rollennamen und einen `SecurityConfiguration`- Namen (d. h., `/aws-glue/jobs-yourRoleName-yourSecurityConfigurationName/`) verwenden, dann wird die Sicherheitskonfiguration verwendet, um die Protokollgruppe zu verschlüsseln.

- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsausführungs-Benachrichtigung an.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  `GlueVersion`Ermittelt in Spark-Jobs die Versionen von Apache Spark und Python, die in einem Job AWS Glue verfügbar sind. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird.

  In Ray-Aufträge sollte die `GlueVersion` auf `4.0` oder höher eingestellt sein. Welche Versionen von Ray, Python und zusätzlichen Bibliotheken in Ihrem Ray-Auftrag verfügbar sind, wird jedoch durch die `Runtime`-Parameter des Auftragsbefehls bestimmt.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

  Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

- `DPUSeconds` – Nummer (doppelt).

  Dieses Feld kann entweder für Jobausführungen mit Ausführungsklasse `FLEX` oder bei aktiviertem Auto Scaling festgelegt werden. Es stellt die Gesamtzeit, die jeder Executor während des Lebenszyklus einer Jobausführung ausgeführt hat, in Sekunden dar, multipliziert mit einem DPU-Faktor (1 für`G.1X`, 2 für oder 0,25 für `G.2X` Worker). `G.025X` Dieser Wert kann von `executionEngineRuntime` * `MaxCapacity` abweichen, da die Anzahl der zu einem bestimmten Zeitpunkt ausgeführten Executors bei Auto-Scaling-Aufträgen geringer sein kann als die `MaxCapacity`. Daher ist es möglich, dass der Wert von `DPUSeconds` kleiner ist als `executionEngineRuntime` * `MaxCapacity`.

- `ExecutionClass` – UTF-8-Zeichenfolge, nicht länger als 16 Bytes (gültige Werte: `FLEX=""` | `STANDARD=""`).

  Gibt an, ob der Auftrag mit einer Standard- oder einer flexiblen Ausführungsklasse ausgeführt wird. Die Standardausführungsklasse ist ideal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

  Die flexible Ausführungsklasse ist geeignet für zeitunabhängige Aufträge, deren Start- und Abschlusszeiten variieren können.

Nur Jobs mit AWS Glue Version 3.0 und höher und dem Befehlstyp dürfen auf `glueetl` gesetzt werden. `ExecutionClass` `FLEX` Die flexible Ausführungsklasse ist für Spark-Aufträge verfügbar.

- `MaintenanceWindow` – UTF-8-Zeichenfolge, die Custom string pattern #34 entspricht.

  Dieses Feld gibt einen Wochentag und eine Stunde für ein Wartungsfenster für Streaming-Jobs an. AWS Glue führt regelmäßig Wartungsarbeiten durch. Während dieser Wartungsfenster müssen AWS Glue Sie Ihre Streaming-Jobs neu starten.

  AWS Glue wird den Job innerhalb von 3 Stunden nach Ablauf des angegebenen Wartungsfensters neu starten. Wenn Sie beispielsweise das Wartungsfenster für Montag um 10:00 Uhr GMT einrichten, werden Ihre Jobs zwischen 10:00 Uhr GMT und 13:00 Uhr GMT neu gestartet.

- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines AWS Glue Nutzungsprofils, das mit dem ausgeführten Job verknüpft ist.

- `StateDetail` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Dieses Feld enthält Details, die sich auf den Status eines ausgeführten Jobs beziehen. Das Feld ist nullwertfähig.

  Wenn sich ein ausgeführter Auftrag beispielsweise aufgrund einer Warteschlange in der Warteschlange befindet, enthält das Feld den Grund, warum sich der ausgeführte Auftrag in diesem Status befindet.

- `ExecutionRoleSessionPolicy`— UTF-8-Zeichenfolge, nicht weniger als 2 oder mehr als 2048 Byte lang.

  Diese Inline-Sitzungsrichtlinie für die StartJobRun API ermöglicht es Ihnen, die Berechtigungen der angegebenen Ausführungsrolle für den Umfang des Auftrags dynamisch einzuschränken, ohne dass zusätzliche IAM-Rollen erstellt werden müssen.

Vorgängerstruktur

Eine Auftragsausführung, die im Prädikat eines bedingten Auslösers verwendet wurde, der diese Auftragsausführung ausgelöst hat.

Felder

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, der bei der vorherigen Auftragsausführung verwendet wurde.
- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftragsausführungs-ID der Vorgängerauftragsausführung.

JobBookmarkEntry Struktur

Definiert einen Punkt, bei dem ein Auftrag die Verarbeitung fortsetzen kann.

Felder

- `JobName` – UTF-8-Zeichenfolge.

  Der Name des jeweiligen Auftrags.
- `Version` – Zahl (Ganzzahl).

  Die Version des Auftrags.
- `Run` – Zahl (Ganzzahl).

  Die Ausführungs-ID-Nummer.
- `Attempt` – Zahl (Ganzzahl).

  Die Versuchs-ID-Nummer.
- `PreviousRunId` – UTF-8-Zeichenfolge.

  Die eindeutige Ausführungskennung, die der vorherigen Auftragsausführung zugeordnet ist.
- `RunId` – UTF-8-Zeichenfolge.

  Die Ausführungs-ID-Nummer.
- `JobBookmark` – UTF-8-Zeichenfolge.

  Die Textmarke selbst.

BatchStopJobRunSuccessfulSubmission Struktur

Zeichnet eine erfolgreiche Anforderung auf, um einen bestimmten JobRun zu stoppen.

Felder

- JobName – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, der bei der gestoppten Auftragsausführung verwendet wurde.
- JobRunId – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die JobRunId der Auftragsausführung, die gestoppt wurde.

BatchStopJobRunError Struktur

Zeichnet einen Fehler auf, der beim Versuch, eine bestimmte Auftragsausführung zu stoppen, aufgetreten ist.

Felder

- JobName – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, der bei der fraglichen Auftragsausführung verwendet wurde.
- JobRunId – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die JobRunId der jeweiligen Auftragsausführung.
- ErrorDetail – Ein ErrorDetail-Objekt.

  Gibt Details über den aufgetretenen Fehler an.

NotificationProperty Struktur

Gibt die Konfigurationseigenschaften einer Benachrichtigung an.

Felder

- `NotifyDelayAfter` – Zahl (ganze Zahl), mindestens 1.

  Nach dem Start eines Auftragslaufs gibt dies die Anzahl der Minuten an, die gewartet werden muss, bevor eine Benachrichtigung über die Verzögerung eines Auftragslaufs gesendet wird.

Operationen

- StartJobRun Aktion (Python: start_job_run)
- BatchStopJobRun Aktion (Python: batch_stop_job_run)
- GetJobRun Aktion (Python: get_job_run)
- GetJobRuns Aktion (Python: get_job_runs)
- GetJobBookmark Aktion (Python: get_job_bookmark)
- GetJobBookmarks Aktion (Python: get_job_bookmarks)
- ResetJobBookmark Aktion (Python: reset_job_bookmark)

StartJobRun Aktion (Python: start_job_run)

Startet einen Auftrag mit einer Auftragsdefinition.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu verwendenden Auftragsdefinition.

- `JobRunQueuingEnabled` – Boolesch.

  Gibt an, ob die Warteschlange für die Auftragsausführung aktiviert ist.

  Der Wert true bedeutet, dass die Warteschlange für die Auftragsausführung aktiviert ist. Wenn der Wert falsch oder nicht ausgefüllt ist, wird der ausgeführte Job nicht in die Warteschlange aufgenommen.

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die ID einer vorherigen `JobRun` für einen erneuten Versuch.

- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die mit dieser Ausführung verknüpften Auftragsargumente. Für diese Auftragsauführung ersetzen sie den Satz der Standardargumente direkt in der Auftragsdefinition.

  Sie können hier Argumente angeben, die Ihr eigenes Jobausführungsskript verwendet, sowie Argumente, die selbst verwendet werden. AWS Glue

  Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

  Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Spark-Aufträgen angeben können, finden Sie im Thema Spezielle Parameter, die von AWS Glue verwendet werden im Entwicklerhandbuch.

  Informationen zu den Argumenten, die Sie für dieses Feld beim Konfigurieren von Ray-Aufträgen angeben können, finden Sie unter Verwenden von Auftragsparametern in Ray-Aufträgen im Entwicklerhandbuch.

- `AllocatedCapacity` – Zahl (Ganzzahl).

  Diese Feld ist als veraltet gekennzeichnet. Verwenden Sie stattdessen `MaxCapacity`.

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem JobRun zugewiesen werden sollen. Sie können mindestens 2 zuweisen DPUs; die Standardeinstellung ist 10. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

DerJobRun-Timeoutwert in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den TIMEOUT-Status wechselt. Dieser Wert überschreibt den Wert der Zeitüberschreitung im übergeordneten Auftrag.

Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `MaxCapacity` – Nummer (doppelt).

  Für Glue-Jobs der Version 1.0 oder früher, die den Standard-Worker-Typ verwenden, die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn dieser Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Bei Aufträgen ab Glue-Version 2.0 können Sie keine `Maximum capacity` angeben. Stattdessen sollten Sie einen `Worker type` und die `Number of workers` festlegen.

  Setzen Sie nicht `MaxCapacity`, wenn Sie `WorkerType` und `NumberOfWorkers` verwenden.

  Der Wert, der `MaxCapacity` zugewiesen werden kann, ist davon abhängig, ob Sie eine Python-Shell-Aufgabe, eine Apache-Spark-ETL-Aufgabe oder eine Apache-Spark-Streaming-ETL-Aufgabe ausführen:

  - Wenn Sie einen Python-Shell-Auftrag (`JobCommand.Name`="pythonshell") angeben, können Sie entweder 0,0625 oder 1 DPU zuweisen. Der Standardwert ist 0,0625 DPU.
  - Wenn Sie einen Apache Spark-ETL-Job (`JobCommand.Name`="glueetl“) oder einen Apache Spark-Streaming-ETL-Job (`JobCommand.Name`="gluestreaming“) angeben, können Sie zwischen 2 und 100 zuweisen. DPUs Die Standardeinstellung ist 10. DPUs Diesem Auftragstyp dürfen keine DPU-Bruchteile zugeteilt werden.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem Auftrag verwendet werden soll.

- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsausführungs-Benachrichtigung an.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird. Akzeptiert einen Wert von G.1X, G.2X, G.4X, G.8X oder G.025X für Spark-Aufträge. Akzeptiert den Wert Z.2X für Ray-Aufträge.

  - Beim `G.1X` Worker-Typ wird jeder Worker einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 94 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.2X` Worker-Typ wird jeder Worker 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 138 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.4X` Worker-Typ wird jeder Worker 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Workertyp ist nur für Spark ETL-Jobs der AWS Glue Version 3.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland) und Europa (Stockholm).
  - Für den `G.8X` Worker-Typ wird jeder Worker 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Jobs der AWS Glue Version 3.0 oder höher in denselben AWS Regionen verfügbar, die auch für den Worker-Typ unterstützt werden. `G.4X`

  - Für den `G.025X` Worker-Typ entspricht jeder Worker 0,25 DPU (2 VCPUs, 4 GB Arbeitsspeicher) mit 84 GB Festplatte und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Streaming-Aufträge mit geringem Volumen. Dieser Worker-Typ ist nur für Streaming-Jobs der AWS Glue Version 3.0 oder höher verfügbar.
  - Für den `Z.2X` Worker-Typ wird jeder Worker 2 M-DPU (8 VCPUs, 64 GB Arbeitsspeicher) mit 128 GB Festplatte zugeordnet und bietet auf Basis des Autoscalers bis zu 8 Ray-Worker.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die zugewiesenj werden, wenn ein Auftrag ausgeführt wird.
- `ExecutionClass` – UTF-8-Zeichenfolge, nicht länger als 16 Bytes (gültige Werte: `FLEX=""` | `STANDARD=""`).

  Gibt an, ob der Auftrag mit einer Standard- oder einer flexiblen Ausführungsklasse ausgeführt wird. Die Standardausführungsklasse ist ideal für zeitkritische Workloads, die einen schnellen Auftragsstart und dedizierte Ressourcen erfordern.

  Die flexible Ausführungsklasse ist geeignet für zeitunabhängige Aufträge, deren Start- und Abschlusszeiten variieren können.

  Nur Jobs mit AWS Glue Version 3.0 und höher und mit dem Befehlstyp dürfen auf `glueetl` gesetzt werden. `ExecutionClass` `FLEX` Die flexible Ausführungsklasse ist für Spark-Aufträge verfügbar.
- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines AWS Glue Verwendungsprofils, das mit der Auftragsausführung verknüpft ist.
- `ExecutionRoleSessionPolicy`— UTF-8-Zeichenfolge, nicht weniger als 2 oder mehr als 2048 Byte lang.

  Diese Inline-Sitzungsrichtlinie für die StartJobRun API ermöglicht es Ihnen, die Berechtigungen der angegebenen Ausführungsrolle für den Umfang des Auftrags dynamisch einzuschränken, ohne dass zusätzliche IAM-Rollen erstellt werden müssen.

Antwort

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Auftragsausführung zugewiesenen ID.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

BatchStopJobRun Aktion (Python: batch_stop_job_run)

Beendet eine oder mehrere Auftragsausführungen für eine bestimmte Auftragsdefinition.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Auftragsdefinition, für die Auftragsausführungen gestoppt werden sollen.
- `JobRunIds` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste der `JobRunIds`, die für diese Auftragsdefinition gestoppt werden sollen.

Antwort

- `SuccessfulSubmissions` – Ein Array mit BatchStopJobRunSuccessfulSubmission-Objekten.

  Eine Liste der, die erfolgreich zum Stoppen eingereicht wurden JobRuns .
- `Errors` – Ein Array mit BatchStopJobRunError-Objekten.

Eine Liste der Fehler, die beim Versuch, die `JobRuns` zu stoppen, aufgetreten sind, einschließlich der `JobRunId`, bei der jeder Fehler aufgetreten ist und Details über den Fehler.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRun Aktion (Python: get_job_run)

Ruft die Metadaten für eine bestimmte Auftragsausführung ab. Der Verlauf der Auftragsausführung ist für Ihren Workflow und Ihre Auftragsausführung 365 Tage lang zugänglich.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der ausgeführten Auftragsdefinition.
- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Auftragsausführung.
- `PredecessorsIncluded` – Boolesch.

  "True", wenn eine Liste der Vorgängerausführungen zurückgegeben werden soll.

Antwort

- `JobRun` – Ein JobRun-Objekt.

  Die angeforderten Metadaten der Auftragsausführung.

Fehler

- `InvalidInputException`

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRuns Aktion (Python: get_job_runs)

Ruft Metadaten für alle Ausführungen einer bestimmten Auftragsdefinition ab.

`GetJobRuns`gibt die Jobläufe in chronologischer Reihenfolge zurück, wobei die neuesten Jobs zuerst zurückgegeben werden.

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name de Auftragsdefinition, für die alle Auftragsausführungen abgerufen werden sollen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.
- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 200.

  Die maximale Größe der Antwort.

Antwort

- `JobRuns` – Ein Array mit JobRun-Objekten.

  Eine Liste der Metadatenobjekte der Auftragsausführung.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, falls nicht alle angeforderten Auftragsausführungen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`

GetJobBookmark Aktion (Python: get_job_bookmark)

Gibt Informationen zu einem Auftrags-Lesezeicheneintrag zurück.

Weitere Informationen zum Aktivieren und Verwenden von Auftragslesezeichen finden Sie unter:

- Verfolgen von verarbeiteten Daten mit Auftragslesezeichen
- Job-Parameter, die verwendet werden von AWS Glue
- Auftrags-Struktur

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name des jeweiligen Auftrags.
- `Version` – Zahl (Ganzzahl).

  Die Version des Auftrags.
- `RunId` – UTF-8-Zeichenfolge.

  Die dieser Auftragsausführung zugeordnete eindeutige Kennung.

Antwort

- `JobBookmarkEntry` – Ein JobBookmarkEntry-Objekt.

  Eine Struktur, die einen Punkt definiert, an dem ein Auftrag die Verarbeitung fortsetzen kann.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

- `ValidationException`

GetJobBookmarks Aktion (Python: get_job_bookmarks)

Gibt Informationen zu den Auftrags-Lesezeicheneinträgen zurück. Die Liste wird nach abnehmenden Versionsnummern sortiert.

Weitere Informationen zum Aktivieren und Verwenden von Auftragslesezeichen finden Sie unter:

- Verfolgen von verarbeiteten Daten mit Auftragslesezeichen
- Job-Parameter, die verwendet werden von AWS Glue
- Auftrags-Struktur

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name des jeweiligen Auftrags.
- `MaxResults` – Zahl (Ganzzahl).

  Die maximale Größe der Antwort.
- `NextToken` – Zahl (Ganzzahl).

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `JobBookmarkEntries` – Ein Array mit JobBookmarkEntry-Objekten.

  Eine Liste der Auftrags-Lesezeicheneinträge, die einen Punkt definiert, an dem ein Auftrag die Verarbeitung fortsetzen kann.
- `NextToken` – Zahl (Ganzzahl).

  Ein Fortsetzungstoken, der den Wert 1 hat, wenn alle Einträge zurückgegeben werden, oder >1, wenn nicht alle angeforderten Auftragsausführungen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ResetJobBookmark Aktion (Python: reset_job_bookmark)

Setzt einen Textmarkeneintrag zurück.

Weitere Informationen zum Aktivieren und Verwenden von Auftragslesezeichen finden Sie unter:

- Verfolgen von verarbeiteten Daten mit Auftragslesezeichen
- Job-Parameter, die verwendet werden von AWS Glue
- Auftrags-Struktur

Anforderung

- `JobName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name des jeweiligen Auftrags.
- `RunId` – UTF-8-Zeichenfolge.

  Die dieser Auftragsausführung zugeordnete eindeutige Kennung.

Antwort

- `JobBookmarkEntry` – Ein JobBookmarkEntry-Objekt.

  Der zurückgesetzte Textmarkeneintrag.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

- `OperationTimeoutException`

Auslöser

Die Trigger-API beschreibt die Datentypen und die API im Zusammenhang mit dem Erstellen, Aktualisieren oder Löschen sowie dem Starten und Stoppen von Job-Triggern in AWS Glue.

Datentypen

- Auslöserstruktur
- TriggerUpdate Struktur
- Prädikatstruktur
- Bedingungsstruktur
- Aktionsstruktur
- EventBatchingCondition Struktur

Auslöserstruktur

Informationen zu einem bestimmten Auslöser.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers.
- `WorkflowName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows, der dem Auslöser zugeordnet ist.
- `Id` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Für die spätere Verwendung reserviert.
- `Type` – UTF-8-Zeichenfolge (zulässige Werte: `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

Der Typ des Auslösers, der hier vorliegt.

- `State` – UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `CREATED` | `ACTIVATING` | `ACTIVATED` | `DEACTIVATING` | `DEACTIVATED` | `DELETING` | `UPDATING`).

  Der aktuelle Status des Auslösers.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung dieses Auslösers.

- `Schedule` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

- `Actions` – Ein Array mit Aktion-Objekten.

  Die durch diesen Auslöser initiierten Aktionen.

- `Predicate` – Ein Prädikat-Objekt.

  Das Prädikat dieses Auslösers, das definiert, wann er ausgelöst wird.

- `EventBatchingCondition` – Ein EventBatchingCondition-Objekt.

  Batch-Bedingung, die erfüllt sein muss (angegebene Anzahl empfangener Ereignisse oder Ablauf des Batchzeitfensters), bevor der EventBridge Ereignisauslöser ausgelöst wird.

TriggerUpdate Struktur

Eine Struktur, die verwendet wird, um Informationen bereitzustellen, mit deren Hilfe ein Auslöser aktualisiert wird. Dieses Objekt aktualisiert die vorherige Auslöserdefinition, indem es sie komplett überschreibt.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Für die spätere Verwendung reserviert.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung dieses Auslösers.

- `Schedule` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

- `Actions` – Ein Array mit Aktion-Objekten.

  Die durch diesen Auslöser initiierten Aktionen.

- `Predicate` – Ein Prädikat-Objekt.

  Das Prädikat dieses Auslösers, das definiert, wann er ausgelöst wird.

- `EventBatchingCondition` – Ein EventBatchingCondition-Objekt.

  Batch-Bedingung, die erfüllt sein muss (angegebene Anzahl empfangener Ereignisse oder Ablauf des Batchzeitfensters), bevor der EventBridge Ereignisauslöser ausgelöst wird.

Prädikatstruktur

Definiert das Prädikat des Auslösers, mit dem festgelegt wird, wann der Auslöser ausgelöst wird.

Felder

- `Logical` – UTF-8-Zeichenfolge (zulässige Werte: `AND` | `ANY`).

  Optionales Feld, wenn nur eine Bedingung aufgeführt ist. Wenn mehrere Bedingungen aufgelistet sind, dann ist dieses Feld erforderlich.

- `Conditions`— Eine Reihe von Bedingung Objekten, nicht mehr als 500 Strukturen.

  Eine Liste der Bedingungen, die bestimmen, wann der Auslöser ausgelöst wird.

Bedingungsstruktur

Definiert die Bedingung, unter der ein Auslöser ausgelöst wird.

Felder

- `LogicalOperator` – UTF-8-Zeichenfolge (zulässige Werte: `EQUALS`).

  Ein logischer Operator.

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auftrags, für dessen `JobRuns` diese Bedingung gilt und auf den dieser Auslöser wartet.

- `State`— UTF-8-String (gültige Werte: `STARTING` | | `RUNNING` | | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` | `ERROR` `WAITING` |`EXPIRED`).

  Der Zustand der Bedingung. Die einzigen Aufgabenstatus, auf die ein Auslöser achten kann, sind `SUCCEEDED`, `STOPPED`, `FAILED` und `TIMEOUT`. Die einzigen Crawler-Status, auf die ein Auslöser achten kann, sind `SUCCEEDED`, `FAILED` und `CANCELLED`.

- `CrawlerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Crawlers, für den die Bedingung gilt.

- `CrawlState` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  Der Status des Crawlers, auf den diese Bedingung angewendet wird.

Aktionsstruktur

Definiert eine Aktion, die von einem Auslöser initiiert werden soll.

Felder

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des auszuführenden Auftrags.

- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

Jeder Wert ist eine UTF-8-Zeichenfolge.

Die Auftragsargumente, die verwendet werden, wenn dieser Auslöser ausgelöst wird. Für diese Auftragsausführung ersetzen sie den Satz der Standardargumente direkt in der Auftragsdefinition.

Sie können hier Argumente angeben, die Ihr eigenes Job-Ausführungsskript verwendet, sowie Argumente, die selbst verwendet werden. AWS Glue

Informationen dazu, wie Sie Ihre eigenen Job-Argumente angeben und verwenden können, finden Sie im Thema Calling AWS Glue APIs in Python im Developer Guide.

Informationen zu den Schlüssel-Wert-Paaren, die für AWS Glue die Einrichtung Ihres Jobs verwendet werden, finden Sie im Entwicklerhandbuch im AWS Glue Thema Spezielle Parameter, die von verwendet werden.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Der`JobRun`-Timeoutwert in Minuten. Dies ist die maximale Zeitspanne, in der eine Auftragsausführung Ressourcen verbrauchen kann, ehe sie beendet wird und in den `TIMEOUT`-Status wechselt. Dadurch wird der Wert der Zeitüberschreitung im übergeordneten Auftrag überschrieben.

  Jobs müssen Timeout-Werte von weniger als 7 Tagen oder 10080 Minuten haben. Andernfalls lösen die Jobs eine Ausnahme aus.

  Wenn der Wert leer gelassen wird, ist das Timeout standardmäßig auf 2880 Minuten festgelegt.

  Für alle vorhandenen AWS Glue Jobs mit einem Timeout-Wert von mehr als 7 Tagen wird standardmäßig ein Wert von 7 Tagen verwendet. Wenn Sie beispielsweise ein Timeout von 20 Tagen für einen Batch-Job angegeben haben, wird dieser am 7. Tag gestoppt.

  Wenn Sie für Streaming-Jobs ein Wartungsfenster eingerichtet haben, wird es während des Wartungsfensters nach 7 Tagen neu gestartet.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit dieser Aktion verwendet werden soll.

- `NotificationProperty` – Ein NotificationProperty-Objekt.

  Gibt die Konfigurationseigenschaften einer Auftragsausführungs-Benachrichtigung an.

- `CrawlerName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Crawlers, der mit dieser Aktion verwendet werden soll.

EventBatchingCondition Struktur

Batch-Bedingung, die erfüllt sein muss (angegebene Anzahl empfangener Ereignisse oder Ablauf des Batchzeitfensters), bevor der EventBridge Ereignisauslöser ausgelöst wird.

Felder

- `BatchSize` – Erforderlich:Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 100.

  Anzahl der Ereignisse, die von Amazon empfangen werden müssen, EventBridge bevor der EventBridge Ereignisauslöser ausgelöst wird.
- `BatchWindow` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 900.

  Zeitfenster in Sekunden, nach dem der EventBridge Ereignisauslöser ausgelöst wird. Das Fenster beginnt, wenn das erste Ereignis empfangen wird.

Operationen

- CreateTrigger Aktion (Python: create_trigger)
- StartTrigger Aktion (Python: start_trigger)
- GetTrigger Aktion (Python: get_trigger)
- GetTriggers Aktion (Python: get_triggers)
- UpdateTrigger Aktion (Python: update_trigger)
- StopTrigger Aktion (Python: stop_trigger)
- DeleteTrigger Aktion (Python: delete_trigger)
- ListTriggers Aktion (Python: list_triggers)
- BatchGetTriggers Aktion (Python: batch_get_triggers)

CreateTrigger Aktion (Python: create_trigger)

Erstellt einen neuen Auslöser.

Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers.
- `WorkflowName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows, der dem Auslöser zugeordnet ist.
- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

  Der Typ des neuen Auslösers.
- `Schedule` – UTF-8-Zeichenfolge.

  Ein `cron`-Ausdruck, der verwendet wird, um den Zeitplan festzulegen (siehe Zeitbasierte Pläne für Aufträge und Crawler). Wenn Sie beispielsweise etwas täglich um 12:15 UTC ausführen möchten, würden Sie Folgendes angeben: `cron(15 12 * * ? *)`.

  Dieses Feld ist erforderlich, wenn der Auslösertyp SCHEDULED ist.
- `Predicate` – Ein Prädikat-Objekt.

  Ein Prädikat, um anzugeben, wann der neue Auslöser ausgelöst werden soll.

  Dieses Feld ist erforderlich, wenn der Auslösertyp `CONDITIONAL` ist.
- `Actions` – Erforderlich: Ein Array mit Aktion-Objekten.

  Die von diesem Auslöser initiierten Aktionen, wenn er ausgelöst wird.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des neuen Auslösers.
- `StartOnCreation` – Boolesch.

Legen Sie den Wert auf `true` fest, um die Auslöser `SCHEDULED` und `CONDITIONAL` zu starten, wenn sie erstellt sind. „True“ wird für `ON_DEMAND`-Auslöser nicht unterstützt.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die mit diesem Auslöser zu verwendenden Tags. Sie können Tags verwenden, um den Zugriff auf den Auslöser einzuschränken. Weitere Informationen zu Tags in AWS Glue finden Sie unter AWS Tags in AWS Glue im Entwicklerhandbuch.

- `EventBatchingCondition` – Ein EventBatchingCondition-Objekt.

  Batch-Bedingung, die erfüllt sein muss (angegebene Anzahl empfangener Ereignisse oder Ablauf des Batchzeitfensters), bevor der EventBridge Ereignisauslöser ausgelöst wird.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers.

Fehler

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

StartTrigger Aktion (Python: start_trigger)

Startet einen vorhandenen Auslöser. Informationen darüber, wie verschiedene Arten von Auslösern gestartet werden, finden Sie im Artikel über das Auslösen von Aufträgen.

Anforderung

- Name – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des zu startenden Auslösers.

Antwort

- Name – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der gestartet wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

GetTrigger Aktion (Python: get_trigger)

Ruft die Definition eines Auslösers ab.

Anforderung

- Name – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der abgerufen werden soll.

Antwort

- `Trigger` – Ein Auslöser-Objekt.

  Die angeforderte Auslöserdefinition.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetTriggers Aktion (Python: get_triggers)

Ruft alle einem Auftrag zugeordneten Auslöser ab.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.
- `DependentJobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auftrags, für den Auslöser abgerufen werden. Der Auslöser, der diesen Auftrag starten kann, wird zurückgegeben. Wenn es keinen solchen Auslöser gibt, werden alle Auslöser zurückgegeben.
- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 200.

  Die maximale Größe der Antwort.

Antwort

- `Triggers` – Ein Array mit Auslöser-Objekten.

  Eine Liste der Auslöser für den angegebenen Auftrag.
- `NextToken` – UTF-8-Zeichenfolge.

Ein Fortsetzungstoken, falls noch nicht alle angeforderten Auslöser zurückgegeben wurden.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

UpdateTrigger Aktion (Python: update_trigger)

Aktualisiert eine Auslöserdefinition.

Auftragsargumente können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Argumente. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie beabsichtigen, sie innerhalb des Job zu behalten.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der aktualisiert werden soll.
- `TriggerUpdate` – Erforderlich: Ein TriggerUpdate-Objekt.

  Die neuen Werte, mit denen der Auslöser aktualisiert werden soll.

Antwort

- `Trigger` – Ein Auslöser-Objekt.

  Die resultierende Auslöserdefinition.

Fehler

- `InvalidInputException`

- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

StopTrigger Aktion (Python: stop_trigger)

Stoppt einen angegebene Auslöser.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der gestoppt werden soll.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der gestoppt wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

DeleteTrigger Aktion (Python: delete_trigger)

Löscht einen angegebenen Auslöser. Wenn der Auslöser nicht gefunden wird, wird keine Ausnahme ausgelöst.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der gelöscht werden soll.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Auslösers, der gelöscht wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

ListTriggers Aktion (Python: list_triggers)

Ruft die Namen aller Trigger-Ressourcen in diesem AWS Konto oder der Ressourcen mit dem angegebenen Tag ab. Mit dieser Operation können Sie sehen, welche Ressourcen in Ihrem Konto verfügbar sind, sowie deren Namen.

Diese Operation akzeptiert das optionale `Tags`-Feld, das Sie als Filter für die Antwort verwenden können, so dass markierte Ressourcen als Gruppe abgerufen werden können. Wenn Sie die Tag-Filterung verwenden, werden nur Ressourcen mit dem Tag abgerufen.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `DependentJobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name des Auftrags, für den Auslöser abgerufen werden sollen. Der Auslöser, der diesen Auftrag starten kann, wird zurückgegeben Wenn es keinen solchen Auslöser gibt, werden alle Auslöser zurückgegeben.

- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 200.

  Die maximale Größe der auszugebenden Liste.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Gibt an, das nur diese markierten Ressourcen zurückgegeben werden sollen.

Antwort

- `TriggerNames` – Ein UTF-8-Zeichenfolgen-Array.

  Die Namen aller Auslöser im Konto oder der Auslöser mit den angegebenen Tags.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetTriggers Aktion (Python: batch_get_triggers)

Gibt eine Liste von Ressourcen-Metadaten für eine bestimmte Liste von Auslöser-Namen zurück. Nach dem Aufrufen der `ListTriggers`-Operation können Sie diese Operation aufrufen, um auf die

Daten zuzugreifen, für die Ihnen Berechtigungen erteilt wurden. Dieser Vorgang unterstützt alle IAM-Berechtigungen, einschließlich Berechtigungsbedingungen, die Tags verwenden.

Anforderung

- `TriggerNames` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste von Auslösernamen, die von der `ListTriggers`-Operation als Namen zurückgegeben werden können.

Antwort

- `Triggers` – Ein Array mit Auslöser-Objekten.

  Eine Liste der Auslöser-Definitionen.
- `TriggersNotFound` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Namen nicht gefundener Auslöser.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

Integration APIs in AWS Glue

Datentypen

- Struktur der Integration
- IntegrationConfig Struktur
- IntegrationPartition Struktur
- IntegrationError Struktur
- IntegrationFilter Struktur
- InboundIntegration Struktur
- SourceProcessingProperties Struktur

- TargetProcessingProperties Struktur
- SourceTableConfig Struktur
- TargetTableConfig Struktur

Struktur der Integration

Beschreibt eine Zero-ETL-Integration.

Felder

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN für die Quelle der Integration.
- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN für das Ziel der Integration.
- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung für die Integration.
- `IntegrationName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein eindeutiger Name für die Integration.
- `IntegrationArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.
- `KmsKeyId`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN eines KMS-Schlüssels, der für die Verschlüsselung des Kanals verwendet wird.
- `AdditionalEncryptionContext` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

Ein optionaler Satz nicht geheimer Schlüssel-Wert-Paare, der zusätzliche Kontextinformationen für die Verschlüsselung enthält. Dies kann nur bereitgestellt werden, wenn es bereitgestellt wird. `KMSKeyId`

- `Tags` – Ein Array mit Markierung-Objekten.

  Der Ressource zugewiesene Metadaten, die aus einer Liste von Schlüssel-Wert-Paaren bestehen.

- `Status` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Die folgenden Status sind möglich:

  - ERSTELLEN: Die Integration wird gerade erstellt.
  - AKTIV: Die Erstellung der Integration ist erfolgreich.
  - ÄNDERN: Die Integration wird geändert.
  - FEHLGESCHLAGEN: Die Erstellung der Integration schlägt fehl.
  - LÖSCHEN: Die Integration wurde gelöscht.
  - SYNCHRONISIEREN: Die Integration wird synchronisiert.
  - NEEDS_ATTENTION: Die Integration erfordert Aufmerksamkeit, z. B. Synchronisation.

- `CreateTime` – Erforderlich: Zeitstempel.

  Die Uhrzeit, zu der die Integration erstellt wurde, in UTC.

- `IntegrationConfig` – Ein IntegrationConfig-Objekt.

  Mit der Integration verknüpfte Eigenschaften.

- `Errors` – Ein Array mit IntegrationError-Objekten.

  Eine Liste von Fehlern im Zusammenhang mit der Integration.

- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.

IntegrationConfig Struktur

Eigenschaften, die mit der Integration verknüpft sind.

Felder

- `RefreshInterval` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Gibt die Häufigkeit an, mit der CDC-Pulls (Change Data Capture) oder inkrementelle Ladevorgänge ausgeführt werden sollen. Dieser Parameter bietet Flexibilität bei der Anpassung der Aktualisierungsrate an Ihre spezifischen Datenaktualisierungsmuster, Überlegungen zur Systemlast und Ziele zur Leistungsoptimierung. Das Zeitintervall kann zwischen 15 Minuten und 8640 Minuten (sechs Tage) eingestellt werden. Unterstützt derzeit nur die Erstellung von. `RefreshInterval`
- `SourceProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Sammlung von Schlüssel-Wert-Paaren, die zusätzliche Eigenschaften für die Integrationsquelle angeben. Diese Eigenschaften stellen Konfigurationsoptionen bereit, mit denen das Verhalten der ODB-Quelle bei Datenintegrationsvorgängen angepasst werden kann.

IntegrationPartition Struktur

Eine Struktur, die beschreibt, wie Daten auf dem Ziel partitioniert werden.

Felder

- `FieldName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Feldname, der zur Partitionierung von Daten auf dem Ziel verwendet wird. Vermeiden Sie es, Spalten mit eindeutigen Werten für jede Zeile (z. B. '', LastModifiedTimestamp 'SystemModTimeStamp') als Partitionsspalte zu verwenden. Diese Spalten eignen sich nicht für die Partitionierung, da sie eine große Anzahl kleiner Partitionen erstellen, was zu Leistungsproblemen führen kann.
- `FunctionSpec` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Gibt die Funktion an, die zum Partitionieren von Daten auf dem Ziel verwendet wird. Der einzig zulässige Wert für diesen Parameter ist „identity“ (Zeichenfolge). Die Funktion „Identität“ stellt sicher, dass die Datenpartitionierung auf dem Ziel demselben Schema folgt wie die Quelle. Mit anderen Worten, die Partitionierungsstruktur der Quelldaten wird im Zielort beibehalten.
- `ConversionSpec` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

Gibt das Zeitstempelformat der Quelldaten an. Gültige Werte für sind:

- `epoch_sec`- Zeitstempel der Unix-Epoche in Sekunden
- `epoch_milli`- Zeitstempel der Unix-Epoche in Millisekunden
- `iso`- Zeitstempel im ISO 8601-Format

> **Note**
>
> Geben Sie diese Option nur an`ConversionSpec`, wenn Sie auf Zeitstempeln basierende Partitionsfunktionen (Jahr, Monat, Tag oder Stunde) verwenden. AWS Glue Zero-ETL verwendet diesen Parameter, um Quelldaten vor der Partitionierung korrekt in das Zeitstempelformat umzuwandeln.
> Verwenden Sie bei der Partitionsfunktion keine Spalten mit hoher Kardinalität. `identity` Zu den Spalten mit hoher Kardinalität gehören:
>
> - Primärschlüssel
> - Zeitstempelfelder (wie,) `LastModifiedTimestamp` `CreatedDate`
> - Vom System generierte Zeitstempel
>
> Durch die Verwendung von Spalten mit hoher Kardinalität bei der Identitätspartitionierung entstehen viele kleine Partitionen, was die Aufnahmeleistung erheblich beeinträchtigen kann.

IntegrationError Struktur

Ein Fehler im Zusammenhang mit einer Zero-ETL-Integration.

Felder

- `ErrorCode` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Code im Zusammenhang mit diesem Fehler.
- `ErrorMessage`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Eine Meldung mit einer Beschreibung des Fehlers.

IntegrationFilter Struktur

Ein Filter, der beim Aufrufen einer `DescribeIntegrations` Anfrage verwendet werden kann.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Name des Filters
- `Values` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Filterwerten.

InboundIntegration Struktur

Eine Struktur für eine Integration, die Daten in eine Ressource schreibt.

Felder

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Quellressource für die Integration.
- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Zielressource für die Integration.
- `IntegrationArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Zero-ETL-Integration.
- `Status` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Die folgenden Status sind möglich:
  - ERSTELLEN: Die Integration wird erstellt.
  - AKTIV: Die Erstellung der Integration ist erfolgreich.
  - ÄNDERN: Die Integration wird geändert.
  - FEHLGESCHLAGEN: Die Erstellung der Integration schlägt fehl.
  - LÖSCHEN: Die Integration wurde gelöscht.

  - SYNCHRONISIEREN: Die Integration wird synchronisiert.
  - NEEDS_ATTENTION: Die Integration erfordert Aufmerksamkeit, z. B. Synchronisation.
- `CreateTime` – Erforderlich: Zeitstempel.

  Die Uhrzeit, zu der die Integration erstellt wurde, in UTC.
- `IntegrationConfig` – Ein IntegrationConfig-Objekt.

  Mit der Integration verknüpfte Eigenschaften.
- `Errors` – Ein Array mit IntegrationError-Objekten.

  Eine Liste von Fehlern im Zusammenhang mit der Integration.

SourceProcessingProperties Struktur

Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.

Felder

- `RoleArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Die IAM-Rolle für den Zugriff auf die AWS Glue Verbindung.

TargetProcessingProperties Struktur

Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Felder

- `RoleArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Die IAM-Rolle für den Zugriff auf die AWS Glue Datenbank.
- `KmsArn`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN des KMS-Schlüssels, der für die Verschlüsselung verwendet wird.
- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Die AWS Glue Netzwerkverbindung zur Konfiguration des AWS Glue Jobs, der in der Kunden-VPC ausgeführt wird.

- `EventBusArn`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN eines Eventbridge-Event-Busses für den Empfang der Integrationsstatusbenachrichtigung.

SourceTableConfig Struktur

Eigenschaften, die vom Quell-Leg verwendet werden, um Daten aus der Quelle zu verarbeiten.

Felder

- `Fields` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Feldern, die für die Filterung auf Spaltenebene verwendet werden. Wird derzeit nicht unterstützt.
- `FilterPredicate` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Eine Bedingungsklausel, die für die Filterung auf Zeilenebene verwendet wird. Wird derzeit nicht unterstützt.
- `PrimaryKey` – Ein UTF-8-Zeichenfolgen-Array.

  Geben Sie den Primärschlüsselsatz für diese Tabelle an. Wird derzeit auf Anfrage speziell für `EntityOf` SAP-Entitäten unterstützt. Wenden Sie sich an den AWS Support, um diese Funktion verfügbar zu machen.
- `RecordUpdateField` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Inkrementelles Feld auf der Grundlage eines Zeitstempels für den Abruf. Wird derzeit nicht unterstützt.

TargetTableConfig Struktur

Eigenschaften, die vom Ziel-Leg verwendet werden, um die Daten auf dem Ziel zu partitionieren.

Felder

- `UnnestSpec` – UTF-8-Zeichenfolge (zulässige Werte: `TOPLEVEL` | `FULL` | `NOUNNEST`).

  Gibt an, wie verschachtelte Objekte auf Elemente der obersten Ebene reduziert werden. Gültige Werte sind: „TOPLEVEL“, „FULL“ oder „NOUNNEST“.

- `PartitionSpec` – Ein Array mit IntegrationPartition-Objekten.

  Bestimmt das Datei-Layout auf dem Ziel.
- `TargetTableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der optionale Name einer Zieltabelle.

Operationen

- CreateIntegration Aktion (Python: create_integration)
- ModifyIntegration Aktion (Python: modify_integration)
- DescribeIntegrations Aktion (Python: describe_integrations)
- DeleteIntegration Aktion (Python: delete_integration)
- DescribeInboundIntegrations Aktion (Python: describe_inbound_integrations)
- CreateIntegrationTableProperties Aktion (Python: create_integration_table_properties)
- UpdateIntegrationTableProperties Aktion (Python: update_integration_table_properties)
- GetIntegrationTableProperties Aktion (Python: get_integration_table_properties)
- DeleteIntegrationTableProperties Aktion (Python: delete_integration_table_properties)
- CreateIntegrationResourceProperty Aktion (Python: create_integration_resource_property)
- UpdateIntegrationResourceProperty Aktion (Python: update_integration_resource_property)
- GetIntegrationResourceProperty Aktion (Python: get_integration_resource_property)
- UntagResource Aktion (Python: untag_resource)
- ListTagsForResource Aktion (Python: list_tags_for_resource)

CreateIntegration Aktion (Python: create_integration)

Erstellt eine Zero-ETL-Integration im Konto des Anrufers zwischen zwei Ressourcen mit Amazon Resource Names (ARNs): dem und. `SourceArn` `TargetArn`

Anforderung

- `IntegrationName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein eindeutiger Name für eine Integration in. AWS Glue

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Quellressource für die Integration.

- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Zielressource für die Integration.

- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung der Integration.

- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.

- `KmsKeyId`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN eines KMS-Schlüssels, der für die Verschlüsselung des Kanals verwendet wird.

- `AdditionalEncryptionContext` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein optionaler Satz nicht geheimer Schlüssel-Wert-Paare, der zusätzliche Kontextinformationen für die Verschlüsselung enthält. Dies kann nur bereitgestellt werden, wenn es bereitgestellt wird. `KMSKeyId`

- `Tags` – Ein Array mit Markierung-Objekten.

  Der Ressource zugewiesene Metadaten, die aus einer Liste von Schlüssel-Wert-Paaren bestehen.

- `IntegrationConfig` – Ein IntegrationConfig-Objekt.

  Die Konfigurationseinstellungen.

Antwort

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Quellressource für die Integration.

- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

Der ARN der Zielressource für die Integration.

- `IntegrationName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein eindeutiger Name für eine Integration in. AWS Glue

- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung der Integration.

- `IntegrationArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die erstellte Integration.

- `KmsKeyId`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN eines KMS-Schlüssels, der für die Verschlüsselung des Kanals verwendet wird.

- `AdditionalEncryptionContext` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein optionaler Satz nicht geheimer Schlüssel-Wert-Paare, der zusätzliche Kontextinformationen für die Verschlüsselung enthält.

- `Tags` – Ein Array mit Markierung-Objekten.

  Der Ressource zugewiesene Metadaten, die aus einer Liste von Schlüssel-Wert-Paaren bestehen.

- `Status` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Der Status der Integration, die gerade erstellt wird.

  Die folgenden Status sind möglich:

  - ERSTELLEN: Die Integration wird gerade erstellt.
  - AKTIV: Die Erstellung der Integration ist erfolgreich.
  - ÄNDERN: Die Integration wird geändert.
  - FEHLGESCHLAGEN: Die Erstellung der Integration schlägt fehl.

  - LÖSCHEN: Die Integration wurde gelöscht.
  - SYNCHRONISIEREN: Die Integration wird synchronisiert.
  - NEEDS_ATTENTION: Die Integration erfordert Aufmerksamkeit, z. B. Synchronisation.
- `CreateTime` – Erforderlich: Zeitstempel.

  Die Uhrzeit, zu der die Integration erstellt wurde, in UTC.
- `Errors` – Ein Array mit IntegrationError-Objekten.

  Eine Liste von Fehlern im Zusammenhang mit der Integrationserstellung.
- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.
- `IntegrationConfig` – Ein IntegrationConfig-Objekt.

  Die Konfigurationseinstellungen.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `IntegrationConflictOperationFault`
- `IntegrationQuotaExceededFault`
- `KMSKeyNotAccessibleFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `ResourceNumberLimitExceededException`
- `InvalidInputException`

ModifyIntegration Aktion (Python: modify_integration)

Ändert eine Zero-ETL-Integration im Konto des Anrufers.

Anforderung

- `IntegrationIdentifier`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.
- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung der Integration.
- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.
- `IntegrationName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Ein eindeutiger Name für eine Integration in AWS Glue.

Antwort

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Quelle für die Integration.
- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN des Ziels für die Integration.
- `IntegrationName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein eindeutiger Name für eine Integration in. AWS Glue
- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung der Integration.
- `IntegrationArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.
- `KmsKeyId`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

Der ARN eines KMS-Schlüssels, der für die Verschlüsselung des Kanals verwendet wird.

- `AdditionalEncryptionContext` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein optionaler Satz nicht geheimer Schlüssel-Wert-Paare, der zusätzliche Kontextinformationen für die Verschlüsselung enthält.

- `Tags` – Ein Array mit Markierung-Objekten.

  Der Ressource zugewiesene Metadaten, die aus einer Liste von Schlüssel-Wert-Paaren bestehen.

- `Status` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Der Status der Integration wird geändert.

  Die folgenden Status sind möglich:

  - ERSTELLEN: Die Integration wird gerade erstellt.
  - AKTIV: Die Erstellung der Integration ist erfolgreich.
  - ÄNDERN: Die Integration wird geändert.
  - FEHLGESCHLAGEN: Die Erstellung der Integration schlägt fehl.
  - LÖSCHEN: Die Integration wurde gelöscht.
  - SYNCHRONISIEREN: Die Integration wird synchronisiert.
  - NEEDS_ATTENTION: Die Integration erfordert Aufmerksamkeit, z. B. Synchronisation.

- `CreateTime` – Erforderlich: Zeitstempel.

  Die Uhrzeit, zu der die Integration erstellt wurde, in UTC.

- `Errors` – Ein Array mit IntegrationError-Objekten.

  Eine Liste von Fehlern im Zusammenhang mit der Integrationsänderung.

- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `IntegrationConflictOperationFault`
- `InvalidIntegrationStateFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `InvalidStateException`
- `InvalidInputException`

DescribeIntegrations Aktion (Python: describe_integrations)

Die API wird verwendet, um eine Liste von Integrationen abzurufen.

Anforderung

- `IntegrationIdentifier` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.
- `Marker` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Ein Wert, der den Ausgangspunkt für den nächsten Satz von Antwortdatensätzen in einer nachfolgenden Anfrage angibt.
- `MaxRecords` – Zahl (Ganzzahl).

  Die Gesamtzahl der Elemente, die in der Ausgabe zurückgegeben werden sollen.
- `Filters` – Ein Array mit IntegrationFilter-Objekten.

  Eine Liste von Schlüsseln und Werten, um die Ergebnisse herauszufiltern. Unterstützte Schlüssel sind „Status“, "IntegrationName„und"SourceArn“. IntegrationName ist auf nur einen Wert beschränkt.

Antwort

- `Integrations` – Ein Array mit Integration-Objekten.

  Eine Liste von Zero-ETL-Integrationen.
- `Marker` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Ein Wert, der den Ausgangspunkt für den nächsten Satz von Antwortdatensätzen in einer nachfolgenden Anfrage angibt.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

DeleteIntegration Aktion (Python: delete_integration)

Löscht die angegebene Zero-ETL-Integration.

Anforderung

- `IntegrationIdentifier`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.

Antwort

- `SourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der ARN der Quelle für die Integration.
- `TargetArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

Der ARN des Ziels für die Integration.

- `IntegrationName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Ein eindeutiger Name für eine Integration in. AWS Glue

- `Description`— UTF-8-Zeichenfolge, nicht mehr als 1000 Byte lang, entspricht. Custom string pattern #12

  Eine Beschreibung der Integration.

- `IntegrationArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) für die Integration.

- `KmsKeyId`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Der ARN eines KMS-Schlüssels, der für die Verschlüsselung des Kanals verwendet wird.

- `AdditionalEncryptionContext` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Ein optionaler Satz nicht geheimer Schlüssel-Wert-Paare, der zusätzliche Kontextinformationen für die Verschlüsselung enthält.

- `Tags` – Ein Array mit Markierung-Objekten.

  Der Ressource zugewiesene Metadaten, die aus einer Liste von Schlüssel-Wert-Paaren bestehen.

- `Status` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Der Status der Integration, die gelöscht wird.

  Die folgenden Status sind möglich:

  - ERSTELLEN: Die Integration wird gerade erstellt.
  - AKTIV: Die Erstellung der Integration ist erfolgreich.
  - ÄNDERN: Die Integration wird geändert.
  - FEHLGESCHLAGEN: Die Erstellung der Integration schlägt fehl.

  - LÖSCHEN: Die Integration wurde gelöscht.
  - SYNCHRONISIEREN: Die Integration wird synchronisiert.
  - NEEDS_ATTENTION: Die Integration erfordert Aufmerksamkeit, z. B. Synchronisation.
- `CreateTime` – Erforderlich: Zeitstempel.

  Die Uhrzeit, zu der die Integration erstellt wurde, in UTC.
- `Errors` – Ein Array mit IntegrationError-Objekten.

  Eine Liste von Fehlern im Zusammenhang mit der Integration.
- `DataFilter`— UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 2048 Byte lang.

  Wählt Quelltabellen für die Integration mithilfe der Maxwell-Filtersyntax aus.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `IntegrationConflictOperationFault`
- `InvalidIntegrationStateFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `InvalidStateException`
- `InvalidInputException`

DescribeInboundIntegrations Aktion (Python: describe_inbound_integrations)

Gibt eine Liste der eingehenden Integrationen für die angegebene Integration zurück.

Anforderung

- `IntegrationArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Amazon-Ressourcenname (ARN) der Integration.
- `Marker` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Ein Token für den Beginn der Seitennummerierung. Dies ist die Markierung aus einer zuvor gekürzten Antwort.
- `MaxRecords` – Zahl (Ganzzahl).

  Die Gesamtzahl der Elemente, die in der Ausgabe zurückgegeben werden sollen.
- `TargetArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Amazon-Ressourcenname (ARN) der Zielressource in der Integration.

Antwort

- `InboundIntegrations` – Ein Array mit InboundIntegration-Objekten.

  Eine Liste der eingehenden Integrationen.
- `Marker` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Ein Wert, der den Ausgangspunkt für den nächsten Satz von Antwortdatensätzen in einer nachfolgenden Anfrage angibt.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `TargetResourceNotFound`
- `OperationNotSupportedException`
- `EntityNotFoundException`
- `InternalServiceException`

- `InvalidInputException`

CreateIntegrationTableProperties Aktion (Python: create_integration_table_properties)

Diese API wird verwendet, um optionale Override-Eigenschaften für die Tabellen bereitzustellen, die repliziert werden müssen. Diese Eigenschaften können Eigenschaften zum Filtern und Partitionieren für die Quell- und Zieltabellen beinhalten. Um sowohl Quell- als auch Zieleigenschaften festzulegen, muss dieselbe API mit dem AWS Glue Verbindungs-ARN wie `ResourceArn` bei `SourceTableConfig` `TargetTableConfig` bzw. dem AWS Glue Datenbank-ARN wie `ResourceArn` mit aufgerufen werden.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) der Zieltabelle, für die Eigenschaften der Integrationstabelle erstellt werden sollen. Derzeit unterstützt diese API nur das Erstellen von Integrationstabelleneigenschaften für Zieltabellen, und der angegebene ARN sollte der ARN der Zieltabelle im AWS Glue Datenkatalog sein. Die Support für die Erstellung von Integrationstabelleneigenschaften für Quellverbindungen (mithilfe des Verbindungs-ARN) ist noch nicht implementiert und wird in einer future Version hinzugefügt.
- `TableName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Name der Tabelle, die repliziert werden soll.
- `SourceTableConfig` – Ein SourceTableConfig-Objekt.

  Eine Struktur für die Konfiguration der Quelltabelle. In der `SourceTableConfig` Struktur finden Sie eine Liste der unterstützten Quelleigenschaften.
- `TargetTableConfig` – Ein TargetTableConfig-Objekt.

  Eine Struktur für die Zieltabellenkonfiguration.

Antwort

- Keine Antwortparameter.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

UpdateIntegrationTableProperties Aktion (Python: update_integration_table_properties)

Diese API wird verwendet, um optionale Override-Eigenschaften für die Tabellen bereitzustellen, die repliziert werden müssen. Diese Eigenschaften können Eigenschaften zum Filtern und Partitionieren für die Quell- und Zieltabellen beinhalten. Um sowohl Quell- als auch Zieleigenschaften festzulegen, muss dieselbe API mit dem AWS Glue Verbindungs-ARN wie `ResourceArn` bei `SourceTableConfig` `TargetTableConfig` bzw. dem AWS Glue Datenbank-ARN wie `ResourceArn` mit aufgerufen werden.

Die Überschreibung wird für alle Integrationen übernommen, die dieselbe Tabelle `ResourceArn` und die Quelltabelle verwenden.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `TableName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Name der Tabelle, die repliziert werden soll.
- `SourceTableConfig` – Ein SourceTableConfig-Objekt.

  Eine Struktur für die Konfiguration der Quelltabelle.
- `TargetTableConfig` – Ein TargetTableConfig-Objekt.

Eine Struktur für die Zieltabellenkonfiguration.

Antwort

- Keine Antwortparameter.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

GetIntegrationTableProperties Aktion (Python: get_integration_table_properties)

Diese API wird verwendet, um optionale Override-Eigenschaften für die Tabellen abzurufen, die repliziert werden müssen. Diese Eigenschaften können Eigenschaften zum Filtern und Partitionieren für Quell- und Zieltabellen beinhalten.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Amazon-Ressourcenname (ARN) der Zieltabelle, für die die Eigenschaften der Integrationstabelle abgerufen werden sollen. Derzeit unterstützt diese API nur das Abrufen von Eigenschaften für Zieltabellen, und der angegebene ARN sollte der ARN der Zieltabelle im AWS Glue Datenkatalog sein. Die Support für das Abrufen von Integrationstabelleneigenschaften für Quellverbindungen (mithilfe des Verbindungs-ARN) ist noch nicht implementiert und wird in einer future Version hinzugefügt.
- `TableName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

Der Name der Tabelle, die repliziert werden soll.

Antwort

- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Amazon-Ressourcenname (ARN) der Zieltabelle, für die die Eigenschaften der Integrationstabelle abgerufen werden sollen. Derzeit unterstützt diese API nur das Abrufen von Eigenschaften für Zieltabellen, und der angegebene ARN sollte der ARN der Zieltabelle im AWS Glue Datenkatalog sein. Die Support für das Abrufen von Integrationstabelleneigenschaften für Quellverbindungen (mithilfe des Verbindungs-ARN) ist noch nicht implementiert und wird in einer future Version hinzugefügt.
- `TableName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Name der Tabelle, die repliziert werden soll.
- `SourceTableConfig` – Ein SourceTableConfig-Objekt.

  Eine Struktur für die Konfiguration der Quelltabelle.
- `TargetTableConfig` – Ein TargetTableConfig-Objekt.

  Eine Struktur für die Zieltabellenkonfiguration.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

DeleteIntegrationTableProperties Aktion (Python: delete_integration_table_properties)

Löscht die Tabelleneigenschaften, die für die Tabellen erstellt wurden, die repliziert werden müssen.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `TableName`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Name der Tabelle, die repliziert werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

CreateIntegrationResourceProperty Aktion (Python: create_integration_resource_property)

Diese API kann für die Einrichtung `ResourceProperty` der AWS Glue Verbindung (für die Quelle) oder des AWS Glue Datenbank-ARN (für das Ziel) verwendet werden. Diese Eigenschaften können die Rolle für den Zugriff auf die Verbindung oder Datenbank beinhalten. Um sowohl Quell- als

auch Zieleigenschaften festzulegen, muss dieselbe API mit dem AWS Glue Verbindungs-ARN wie `ResourceArn` bei `SourceProcessingProperties` `TargetProcessingProperties` bzw. dem AWS Glue Datenbank-ARN wie `ResourceArn` mit aufgerufen werden.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `SourceProcessingProperties` – Ein SourceProcessingProperties-Objekt.

  Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.
- `TargetProcessingProperties` – Ein TargetProcessingProperties-Objekt.

  Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Antwort

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `SourceProcessingProperties` – Ein SourceProcessingProperties-Objekt.

  Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.
- `TargetProcessingProperties` – Ein TargetProcessingProperties-Objekt.

  Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `ConflictException`
- `InternalServerException`
- `ResourceNotFoundException`

- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

UpdateIntegrationResourceProperty Aktion (Python: update_integration_resource_property)

Diese API kann für die Aktualisierung `ResourceProperty` der AWS Glue Verbindung (für die Quelle) oder des AWS Glue Datenbank-ARN (für das Ziel) verwendet werden. Diese Eigenschaften können die Rolle für den Zugriff auf die Verbindung oder Datenbank beinhalten. Da dieselbe Ressource für mehrere Integrationen verwendet werden kann, wirkt sich die Aktualisierung der Ressourceneigenschaften auf alle Integrationen aus, die sie verwenden.

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `SourceProcessingProperties` – Ein SourceProcessingProperties-Objekt.

  Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.
- `TargetProcessingProperties` – Ein TargetProcessingProperties-Objekt.

  Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Antwort

- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `SourceProcessingProperties` – Ein SourceProcessingProperties-Objekt.

  Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.
- `TargetProcessingProperties` – Ein TargetProcessingProperties-Objekt.

  Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `ResourceNotFoundException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

GetIntegrationResourceProperty Aktion (Python: get_integration_resource_property)

Diese API wird verwendet, um den `ResourceProperty` AWS Glue Verbindungs-ARN (für die Quelle) oder den AWS Glue Datenbank-ARN (für das Ziel) abzurufen

Anforderung

- `ResourceArn`— Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Byte lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.

Antwort

- `ResourceArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Verbindungs-ARN der Quelle oder der Datenbank-ARN des Ziels.
- `SourceProcessingProperties` – Ein SourceProcessingProperties-Objekt.

  Die mit der Integrationsquelle verknüpften Ressourceneigenschaften.
- `TargetProcessingProperties` – Ein TargetProcessingProperties-Objekt.

  Die mit dem Integrationsziel verknüpften Ressourceneigenschaften.

Fehler

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `ResourceNotFoundException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

UntagResource Aktion (Python: untag_resource)

Entfernt die angegebenen Tags aus einer Integrationsressource.

Anforderung

- `ResourceArn` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) für die Integrationsressource.
- `TagsToRemove` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 50 Zeichenfolgen.

  Eine Liste von Metadaten-Tags, die aus der Ressource entfernt werden sollen.

Antwort

- Keine Antwortparameter.

Fehler

- `ResourceNotFoundException`

ListTagsForResource Aktion (Python: list_tags_for_resource)

Listet die Metadaten-Tags auf, die der angegebenen Ressource zugewiesen sind.

Anforderung

- `ResourceARN` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Ressourcen-ARN für die Ressource.

Antwort

- `Tags` – Ein Array mit Markierung-Objekten, nicht mehr als 10 Strukturen.

  Eine Liste von Tags.

Fehler

- `ResourceNotFoundException`

Ausnahmen

- ResourceNotFoundException Struktur
- InternalServerException Struktur
- IntegrationAlreadyExistsFault Struktur
- IntegrationConflictOperationFault Struktur
- IntegrationQuotaExceededFault Struktur
- KMSKeyNotAccessibleFault Struktur
- IntegrationNotFoundFault Struktur
- TargetResourceNotFound Struktur
- InvalidIntegrationStateFault Struktur

ResourceNotFoundException Struktur

Die Ressource konnte nicht gefunden werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

InternalServerException Struktur

Es ist ein interner Serverfehler aufgetreten.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IntegrationAlreadyExistsFault Struktur

Die angegebene Integration ist bereits vorhanden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IntegrationConflictOperationFault Struktur

Die angeforderte Operation steht in Konflikt mit einer anderen Operation.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IntegrationQuotaExceededFault Struktur

Die im Rahmen Ihrer Integration verarbeiteten Daten haben Ihr Kontingent überschritten.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

KMSKeyNotAccessibleFault Struktur

Auf den angegebenen KMS-Schlüssel kann nicht zugegriffen werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IntegrationNotFoundFault Struktur

Die angegebene Integration konnte nicht gefunden werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

TargetResourceNotFound Struktur

Die Zielressource konnte nicht gefunden werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InvalidIntegrationStateFault Struktur

Die Integration befindet sich in einem ungültigen Zustand.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

Interaktive Sitzungs-API

Die API für interaktive Sitzungen beschreibt die AWS Glue API im Zusammenhang mit der Verwendung AWS Glue interaktiver Sitzungen zum Erstellen und Testen von ETL-Skripts (Extrahieren, Transformieren und Laden) für die Datenintegration.

Datentypen

- Sitzungsstruktur
- SessionCommand Struktur
- Statement-Struktur
- StatementOutput Struktur
- StatementOutputData Struktur
- ConnectionsList Struktur

Sitzungsstruktur

Der Zeitraum, in dem eine Remote-Spark-Laufzeitumgebung ausgeführt wird.

Felder

- `Id` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Sitzung.
- `CreatedOn` – Zeitstempel.

  Die Uhrzeit und das Datum, an dem die Sitzung erstellt wurde.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `PROVISIONING` | `READY` | `FAILED` | `TIMEOUT` | `STOPPING` | `STOPPED`).

  Der Sitzungsstatus.
- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

Die während der Sitzung angezeigten Fehlermeldung.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Beschreibung der Sitzung.

- `Role` – UTF-8-Zeichenfolge, nicht weniger als 20 oder mehr als 2 048 Bytes lang, passend zum Custom string pattern #30.

  Der Name oder der Amazon-Ressourcenname (ARN) der IAM-Rolle, die diesem Vortrag zugeordnet ist.

- `Command` – Ein SessionCommand-Objekt.

  Der Befehl Object.see. SessionCommand

- `DefaultArguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 75 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, und entspricht dem Custom string pattern #31.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 4096 Bytes lang, und entspricht dem URI address multi-line string pattern.

  Ein Map-Array von Schlüssel-Wert-Paaren. Maximal 75 Paare.

- `Connections` – Ein ConnectionsList-Objekt.

  Die Anzahl der Verbindungen, die für die Sitzung verwendet werden.

- `Progress` – Nummer (doppelt).

  Der Fortschritt der Codeausführung der Sitzung.

- `MaxCapacity` – Nummer (doppelt).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn der Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher besteht.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der SecurityConfiguration Struktur, die für die Sitzung verwendet werden soll.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Die AWS Glue Version bestimmt, welche Versionen von Apache Spark und Python AWS Glue unterstützt werden. Der GlueVersion muss größer als 2,0 sein.

- `DataAccessId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 36 Bytes lang.

  Die Datenzugriffs-ID der Sitzung.

- `PartitionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 36 Bytes lang.

  Die Partitions-ID der Sitzung.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `WorkerType`, die für die Sitzung verwendet werden sollen.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der bei der Ausführung einer Sitzung zugewiesen wird. Akzeptiert einen Wert von `G.1X`, `G.2X`, `G.4X` oder `G.8X` für Spark-Sitzungen. Akzeptiert den Wert `Z.2X` für Ray-Sitzungen.

- `CompletedOn` – Zeitstempel.

  Datum und Uhrzeit des Abschlusses dieser Sitzung.

- `ExecutionTime` – Nummer (doppelt).

  Die Gesamtzeit, in der die Sitzung ausgeführt wurde.

- `DPUSeconds` – Nummer (doppelt).

  Der während der Sitzung DPUs verbrauchte Wert (Formel: ExecutionTime * MaxCapacity).

- `IdleTimeout` – Zahl (Ganzzahl).

  Die Anzahl der Minuten im Leerlauf, bevor die Sitzung das Zeitlimit überschreitet.

- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines mit der Sitzung verknüpften AWS Glue Nutzungsprofils.

SessionCommand Struktur

Der `SessionCommand`, der den Auftrag ausführt.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen der an SessionCommand. Kann „glueetl“ oder „gluestreaming“ sein.
- `PythonVersion` – UTF-8-Zeichenfolge, die Custom string pattern #48 entspricht.

  Gibt die Version von Python an. Die Python-Version gibt die Version an, die für Aufträge vom Typ Spark unterstützt wird.

Statement-Struktur

Die Anweisung oder Aufforderung, dass eine bestimmte Aktion in einer Sitzung ausgeführt wird.

Felder

- `Id` – Zahl (Ganzzahl).

  Die ID der Anweisung.
- `Code` – UTF-8-Zeichenfolge.

  Der Ausführungscode der Anweisung.
- `State` – UTF-8-Zeichenfolge (zulässige Werte: `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | `ERROR`).

  Der Status, während die Anfrage bearbeitet wird.
- `Output` – Ein StatementOutput-Objekt.

  Die Ausgabe in JSON.
- `Progress` – Nummer (doppelt).

  Der Fortschritt der Codeausführung.
- `StartedOn` – Zahl (lang).

Datum und Datum der Unix-Version, an der die Auftragsdefinition gestartet wurde.

- `CompletedOn` – Zahl (lang).

  Datum und Uhrzeit der Unix-Version, an der die Auftragsdefinition abgeschlossen wurde.

StatementOutput Struktur

Die Ausgabe der Codeausführung im JSON-Format.

Felder

- `Data` – Ein StatementOutputData-Objekt.

  Ausgabe der Codeausführung.
- `ExecutionCount` – Zahl (Ganzzahl).

  Die Ausführungszahl der Ausgabe.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | `ERROR`).

  Der Status der Codeausführungs-Ausgabe.
- `ErrorName` – UTF-8-Zeichenfolge.

  Der Name des Fehlers in der Ausgabe.
- `ErrorValue` – UTF-8-Zeichenfolge.

  Der Fehlerwert der Ausgabe.
- `Traceback` – Ein UTF-8-Zeichenfolgen-Array.

  Das Traceback der Ausgabe.

StatementOutputData Struktur

Die Ausgabe der Codeausführung im JSON-Format.

Felder

- `TextPlain` – UTF-8-Zeichenfolge.

Die Ausgabe der Codeausführung im Textformat.

ConnectionsList Struktur

Gibt die Verbindungen an, die von einem Auftrag verwendet werden.

Felder

- `Connections`— Ein Array von UTF-8-Strings, nicht mehr als 1000 Zeichenketten.

  Eine Liste der Verbindungen, die vom Auftrag verwendet werden.

Operationen

- CreateSession Aktion (Python: create_session)
- StopSession Aktion (Python: stop_session)
- DeleteSession Aktion (Python: delete_session)
- GetSession Aktion (Python: get_session)
- ListSessions Aktion (Python: list_sessions)
- RunStatement Aktion (Python: run_statement)
- CancelStatement Aktion (Python: cancel_statement)
- GetStatement Aktion (Python: get_statement)
- ListStatements Aktion (Python: list_statements)

CreateSession Aktion (Python: create_session)

Erstellt eine neue Sitzung.

Anforderung

Anforderung, eine neue Sitzung zu erstellen.

- `Id` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Sitzungs-Anforderung.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Beschreibung der Sitzung.

- `Role` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 20 oder mehr als 2 048 Bytes lang, passend zum Custom string pattern #30.

  Der IAM-Rollen-ARN

- `Command` – Erforderlich: Ein SessionCommand-Objekt.

  Der `SessionCommand`, der den Auftrag ausführt.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Die Anzahl der Minuten, bevor eine Zeitüberschreitung für die Sitzung auftritt. Die Standardeinstellung für Spark-ETL-Jobs ist 48 Stunden (2880 Minuten). Konsultieren Sie die Dokumentation für andere Auftragstypen.

- `IdleTimeout` – Zahl (ganze Zahl), mindestens 1.

  Die Anzahl der untätigen Minuten, bevor eine Zeitüberschreitung für die Sitzung auftritt. Die Standardeinstellung für Spark-ETL-Aufträge ist der Wert der Zeitüberschreitung. Konsultieren Sie die Dokumentation für andere Auftragstypen.

- `DefaultArguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 75 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, und entspricht dem Custom string pattern #31.

  Jeder Wert ist eine UTF-8-Zeichenfolge, nicht mehr als 4096 Bytes lang, und entspricht dem URI address multi-line string pattern.

  Ein Map-Array von Schlüssel-Wert-Paaren. Maximal 75 Paare.

- `Connections` – Ein ConnectionsList-Objekt.

  Die Anzahl der Verbindungen, die für die Sitzung verwendet werden sollen.

- `MaxCapacity` – Nummer (doppelt).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die zugewiesen werden können, wenn der Job ausgeführt wird. Eine DPU ist ein relatives Maß für die Rechenleistung, das aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher besteht.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `WorkerType`, die für die Sitzung verwendet werden sollen.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ der vordefinierten Worker, der zugeordnet wird, wenn ein Auftrag ausgeführt wird. Akzeptiert einen Wert von G.1X, G.2X, G.4X oder G.8X für Spark-Aufträge. Akzeptiert den Wert Z.2X für Ray-Notebooks.

  - Für den `G.1X` Worker-Typ wird jeder Worker einer DPU (4 VCPUs, 16 GB Arbeitsspeicher) mit 94 GB Festplatte zugeordnet und bietet einen Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.2X` Worker-Typ wird jeder Worker 2 DPU (8 VCPUs, 32 GB Arbeitsspeicher) mit 138 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Workloads wie Datentransformationen, Zusammenführungen und Abfragen. Er bietet eine skalierbare und kostengünstige Möglichkeit, die meisten Aufträge auszuführen.
  - Für den `G.4X` Worker-Typ wird jeder Worker 4 DPU (16 VCPUs, 64 GB Arbeitsspeicher) mit 256 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Workertyp ist nur für Spark ETL-Jobs der AWS Glue Version 3.0 oder höher in den folgenden AWS Regionen verfügbar: USA Ost (Ohio), USA Ost (Nord-Virginia), USA West (Oregon), Asien-Pazifik (Singapur), Asien-Pazifik (Sydney), Asien-Pazifik (Tokio), Kanada (Zentral), Europa (Frankfurt), Europa (Irland) und Europa (Stockholm).
  - Für den `G.8X` Worker-Typ wird jeder Worker 8 DPU (32 VCPUs, 128 GB Arbeitsspeicher) mit 512 GB Festplatte zugeordnet und bietet 1 Executor pro Worker. Wir empfehlen diesen Worker-Typ für Aufträge, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Verknüpfungen und Abfragen enthalten. Dieser Worker-Typ ist nur für Spark-ETL-Jobs der AWS Glue Version 3.0 oder höher in denselben AWS Regionen verfügbar, die auch für den Worker-Typ unterstützt werden. `G.4X`
  - Für den `Z.2X` Worker-Typ wird jeder Worker 2 M-DPU (8 VCPUs, 64 GB Arbeitsspeicher) mit 128 GB Festplatte zugeordnet und bietet auf Basis des Autoscalers bis zu 8 Ray-Worker.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der SecurityConfiguration Struktur, die für die Sitzung verwendet werden soll

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Die AWS Glue Version bestimmt, welche Versionen von Apache Spark und Python AWS Glue unterstützt werden. Der GlueVersion muss größer als 2,0 sein.
- `DataAccessId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 36 Bytes lang.

  Die Datenzugriffs-ID der Sitzung.
- `PartitionId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 36 Bytes lang.

  Die Partitions-ID der Sitzung.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Zuordnung der Schlüsselwertpaare (Tags), die zur Sitzung gehören.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.
- `ProfileName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines mit der Sitzung verknüpften AWS Glue Nutzungsprofils.

Antwort

- `Session` – Ein Sitzung-Objekt.

  Gibt das Sitzungsobjekt in der Antwort zurück.

Fehler

- `AccessDeniedException`

- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`

StopSession Aktion (Python: stop_session)

Hält die Sitzung an.

Anforderung

- `Id` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Sitzung, die angehalten werden soll.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.

Antwort

- `Id` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt die ID der angehaltenen Sitzung zurück.

Fehler

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

- `IllegalSessionStateException`
- `ConcurrentModificationException`

DeleteSession Aktion (Python: delete_session)

Löscht die Sitzung.

Anforderung

- `Id` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der zu löschenden Sitzung.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Ursprungs der Löschsitzungsanforderung.

Antwort

- `Id` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt die ID der gelöschten Sitzung zurück.

Fehler

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`
- `ConcurrentModificationException`

GetSession Aktion (Python: get_session)

Ruft die Sitzung ab.

Anforderung

- `Id` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Sitzung.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.

Antwort

- `Session` – Ein Sitzung-Objekt.

  Das Sitzungsobjekt wird in der Antwort zurückgegeben.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

ListSessions Aktion (Python: list_sessions)

Abrufen einer Liste von Sitzungen.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Das Token für den nächsten Ergebnissatz bzw. Null, wenn keine weiteren Ergebnisse vorliegen.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl von Ergebnissen.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Tags, die zur Sitzung gehören.

- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.

Antwort

- `Ids` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt die ID der Sitzungs-Anforderung zurück.

- `Sessions` – Ein Array mit Sitzung-Objekten.

  Gibt das Sitzungs-Objekt zurück.

- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Das Token für den nächsten Ergebnissatz bzw. Null, wenn keine weiteren Ergebnisse vorliegen.

Fehler

- `AccessDeniedException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

RunStatement Aktion (Python: run_statement)

Führt die Anweisung aus.

Anforderung

- `SessionId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Sitzungs-ID der auszuführenden Anweisung.
- `Code` – Erforderlich: UTF-8-Zeichenfolge, nicht länger als 68 000 Bytes.

  Der auszuführende Anweisungscode.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.

Antwort

- `Id` – Zahl (Ganzzahl).

  Gibt die ID der ausgeführten Anweisung zurück.

Fehler

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`
- `IllegalSessionStateException`

CancelStatement Aktion (Python: cancel_statement)

Bricht die Anweisung ab.

Anforderung

- `SessionId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Sitzungs-ID der Anweisung, die abgebrochen werden soll.
- `Id` – Erforderlich: Zahl (Ganzzahl).

  Die ID der Anweisung, die abgebrochen werde soll.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung, die Anweisung abzubrechen.

Antwort

- Keine Antwortparameter.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

GetStatement Aktion (Python: get_statement)

Ruft die Anweisung ab.

Anforderung

- `SessionId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Sitzungs-ID der Anweisung.
- `Id` – Erforderlich: Zahl (Ganzzahl).

  Die ID der Anweisung.
- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung.

Antwort

- `Statement` – Ein Statement-Objekt.

  Gibt die Anweisung zurück.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

ListStatements Aktion (Python: list_statements)

Listet Anweisungen für die Sitzung auf.

Anforderung

- `SessionId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die Sitzungs-ID der Anweisungen.

- `RequestOrigin` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Ursprung der Anforderung, Anweisungen aufzulisten.

- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `Statements` – Ein Array mit Statement-Objekten.

  Gibt die Liste der Anweisungen zurück.

- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Ein Fortsetzungstoken, wenn noch nicht alle Anweisungen zurückgegeben wurden.

Fehler

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

API für Entwicklungsendpunkte

Die API für Entwicklungsendpunkte beschreibt die AWS Glue API, die sich auf das Testen mit einer benutzerdefinierten DevEndpoint Komponente bezieht.

Datentypen

- DevEndpoint Struktur

- DevEndpointCustomLibraries Struktur

DevEndpoint Struktur

Ein Entwicklungsendpunkt, an dem ein Entwickler Remote Debug-ETL (Extract, Transform and Load)-Skripts remote debuggen kann.

Felder

- `EndpointName` – UTF-8-Zeichenfolge.

  Der Name der `DevEndpoint`.

- `RoleArn` – UTF-8-Zeichenfolge, die AWS IAM ARN string pattern entspricht.

  Der Amazon-Ressourcenname (ARN) der IAM-Rolle für diesen `DevEndpoint`.

- `SecurityGroupIds` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste von Sicherheitsgruppen-Kennungen in diesem `DevEndpoint`.

- `SubnetId` – UTF-8-Zeichenfolge.

  Die Subnetz-ID für diesen `DevEndpoint`.

- `YarnEndpointAddress` – UTF-8-Zeichenfolge.

  Die von diesem `DevEndpoint` verwendete YARN-Endpunktadresse.

- `PrivateAddress` – UTF-8-Zeichenfolge.

  Eine private IP-Adresse für den Zugriff auf den `DevEndpoint` in einer VPC, wenn der `DevEndpoint` innerhalb eines Kontos erstellt wird. Das `PrivateAddress`-Feld ist nur dann vorhanden, wenn Sie den `DevEndpoint` in Ihrer VPC erstellen.

- `ZeppelinRemoteSparkInterpreterPort` – Zahl (Ganzzahl).

  Der Apache Zeppelin-Port für den Apache-Spark-Remote-Interpreter.

- `PublicAddress` – UTF-8-Zeichenfolge.

  Die öffentliche IP-Adresse, die dieser `DevEndpoint` verwendet. Das `PublicAddress`-Feld ist nur dann vorhanden, wenn Sie einen Nicht-Virtual-Private-Cloud-(VPC-)`DevEndpoint` erstellen.

- `Status` – UTF-8-Zeichenfolge.

Der aktuelle Status dieses `DevEndpoint`.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der dem Entwicklungsendpunkt zugeordnet ist. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` ist jedem Worker 1 DPU (4 vCPU, 16 GB Arbeitsspeicher, 64 GB Festplattenplatz) sowie ein Executor pro Worker zugewiesen. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge.
  - Für den Worker-Typ `G.2X` sind jedem Worker 2 DPU (8 vCPU, 32 GB Arbeitsspeicher, 128 GB Fewstplattenplatz) sowie ein Executor pro Worker zugewiesen. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge.

  Bekanntes Problem: Wenn ein Entwicklungsendpunkt mit der Konfiguration `G.2X` `WorkerType` erstellt wird, werden die Spark-Treiber für den Entwicklungsendpunkt auf 4 vCPU, 16 GB Arbeitsspeicher und einem 64 GB-Datenträger ausgeführt.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Die Glue-Version bestimmt die Versionen von Apache Spark und Python, die AWS Glue unterstützt werden. Die Python-Version gibt die Version an, die für die Ausführung Ihrer ETL-Skripts auf Entwicklungsendpunkten unterstützt wird.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

  Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

  Sie können eine Version der Python-Unterstützung für Entwicklungsendpunkte angeben, indem Sie den `Arguments` Parameter im `CreateDevEndpoint` oder `UpdateDevEndpoint` APIs verwenden. Wenn keine Argumente angegeben werden, verwendet die Version standardmäßig Python 2.

- `NumberOfWorkers` – Zahl (Ganzzahl).

Die Anzahl der Worker eines definierten `workerType` die dem Entwicklungsendpunkt zugeordnet sind.

Die maximale Anzahl von Workern, die Sie definieren können, ist 299 für `G.1X` und 149 für `G.2X`.

- `NumberOfNodes` – Zahl (Ganzzahl).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem `DevEndpoint` zugewiesen sind.

- `AvailabilityZone` – UTF-8-Zeichenfolge.

  Die AWS Availability Zone, in `DevEndpoint` der sich das befindet.

- `VpcId` – UTF-8-Zeichenfolge.

  Die ID der von diesem `DevEndpoint`verwendeten Virtual Private Cloud (VPC).

- `ExtraPythonLibsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Python-Bibliothek in einem Amazon S3-Bucket, die in Ihrem `DevEndpoint` geladen werden soll. Mehrere Werte müssen vollständige Pfade sein, die durch Kommata getrennt werden.

  > **Note**
  >
  > Sie können nur reine Python-Bibliotheken mit einem `DevEndpoint` verwenden. Bibliotheken, die auf C-Erweiterungen basieren, wie zum Beispiel die pandas-Python-Datenanalysebibliothek, werden derzeit nicht unterstützt.

- `ExtraJarsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Java-`.jar`-Datei in einem S3-Bucket, die in Ihren `DevEndpoint` geladen werden soll.

  > **Note**
  >
  > Sie können nur reine Java-/Scala-Bibliotheken mit einem `DevEndpoint` verwenden.

- `FailureReason` – UTF-8-Zeichenfolge.

  Der Grund für einen aktuellen Ausfall an diesem `DevEndpoint`.

- `LastUpdateStatus` – UTF-8-Zeichenfolge.

  Der Status des letzten Updates.

- `CreatedTimestamp` – Zeitstempel.

  Der Zeitpunkt, zu dem dies erstellt DevEndpoint wurde.

- `LastModifiedTimestamp` – Zeitstempel.

  Der Zeitpunkt, an dem dieser `DevEndpoint` zuletzt geändert wurde.

- `PublicKey` – UTF-8-Zeichenfolge.

  Der öffentliche Schlüssel, den dieser `DevEndpoint` für die Authentifizierung verwenden soll.. Dieses Attribut wird aus Gründen der Abwärtskompatibilität bereitgestellt, da empfohlene Attribut „öffentliche Schlüssel“ ist.

- `PublicKeys` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 5 Zeichenfolgen.

  Eine Liste der von dem `DevEndpoints` für die Authentifizierung verwendeten öffentlichen Schlüssel. Die Verwendung dieses Attributs wird gegenüber einem einzelnen öffentlichen Schlüssel bevorzugt, da die öffentlichen Schlüssel einen jeweils anderen privaten Schlüssel pro Client erlauben.

  > **Note**
  >
  > Wenn Sie zuvor einen Endpunkt mit einem öffentlichen Schlüssel erstellt haben, müssen Sie diesen Schlüssel entfernen, um eine Liste der öffentlichen Schlüssel erstellen zu können. Rufen Sie die `UpdateDevEndpoint`-API-Operation mit dem öffentlichen Schlüsselinhalt im `deletePublicKeys`-Attribut sowie die Liste der neuen Schlüssel im `addPublicKeys`-Attribut auf.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem `DevEndpoint` verwendet werden soll.

- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

Jeder Wert ist eine UTF-8-Zeichenfolge.

Eine Zuordnung von Argumenten, die zum Konfigurieren des `DevEndpoint`verwendet werden.

Gültige Argumente sind:

- `"--enable-glue-datacatalog": ""`

Sie können eine Version der Python-Unterstützung für Entwicklungsendpunkte angeben, indem Sie den `Arguments` Parameter im `CreateDevEndpoint` oder `UpdateDevEndpoint` APIs verwenden. Wenn keine Argumente angegeben werden, verwendet die Version standardmäßig Python 2.

DevEndpointCustomLibraries Struktur

Benutzerdefinierte Bibliotheken, die in einen Entwicklungsendpunkt geladen werden sollen.

Felder

- `ExtraPythonLibsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Python-Bibliothek in einem Amazon Simple Storage Service (Amazon S3)-Bucket, die in Ihren `DevEndpoint` geladen werden soll. Mehrere Werte müssen vollständige Pfade sein, die durch Kommata getrennt werden.

  > **Note**
  >
  > Sie können nur reine Python-Bibliotheken mit einem `DevEndpoint` verwenden. Bibliotheken, die auf C-Erweiterungen basieren, wie zum Beispiel die pandas-Python-Datenanalysebibliothek, werden derzeit nicht unterstützt.

- `ExtraJarsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Java-`.jar`-Datei in einem S3-Bucket, die in Ihren `DevEndpoint` geladen werden soll.

  > **Note**
  >
  > Sie können nur reine Java-/Scala-Bibliotheken mit einem `DevEndpoint` verwenden.

Operationen

- CreateDevEndpoint Aktion (Python: create_dev_endpoint)
- UpdateDevEndpoint Aktion (Python: update_dev_endpoint)
- DeleteDevEndpoint Aktion (Python: delete_dev_endpoint)
- GetDevEndpoint Aktion (Python: get_dev_endpoint)
- GetDevEndpoints Aktion (Python: get_dev_endpoints)
- BatchGetDevEndpoints Aktion (Python: batch_get_dev_endpoints)
- ListDevEndpoints Aktion (Python: list_dev_endpoints)

CreateDevEndpoint Aktion (Python: create_dev_endpoint)

Erstellt einen neuen Entwicklungsendpunkt.

Anforderung

- `EndpointName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name, der dem neuen `DevEndpoint` zugewiesen werden soll.
- `RoleArn` – Erforderlich: UTF-8-Zeichenfolge, die mit der AWS IAM ARN string pattern übereinstimmt.

  Die IAM-Rolle für den `DevEndpoint`.
- `SecurityGroupIds` – Ein UTF-8-Zeichenfolgen-Array.

  Sicherheitsgruppe IDs für die Sicherheitsgruppen, die von den neuen verwendet werden sollen. `DevEndpoint`
- `SubnetId` – UTF-8-Zeichenfolge.

  Die Subnetz-ID für den neuen `DevEndpoint`.
- `PublicKey` – UTF-8-Zeichenfolge.

  Der öffentliche Schlüssel, den dieser `DevEndpoint` für die Authentifizierung verwenden soll.. Dieses Attribut wird aus Gründen der Abwärtskompatibilität bereitgestellt, da empfohlene Attribut „öffentliche Schlüssel“ ist.
- `PublicKeys` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 5 Zeichenfolgen.

Eine Liste der von Entwicklungsendpunkten für die Authentifizierung zu verwendenden öffentlichen Schlüssel. Die Verwendung dieses Attributs wird gegenüber einem einzelnen öffentlichen Schlüssel bevorzugt, da die öffentlichen Schlüssel einen jeweils anderen privaten Schlüssel pro Client erlauben.

> **Note**
>
> Wenn Sie zuvor einen Endpunkt mit einem öffentlichen Schlüssel erstellt haben, müssen Sie diesen Schlüssel entfernen, um eine Liste der öffentlichen Schlüssel erstellen zu können. Rufen Sie die `UpdateDevEndpoint`-API mit dem öffentlichen Schlüsselinhalt im `deletePublicKeys`-Attribut sowie die Liste der neuen Schlüssel im `addPublicKeys`-Attribut auf.

- `NumberOfNodes` – Zahl (Ganzzahl).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem `DevEndpoint` zugewiesen werden sollen.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der dem Entwicklungsendpunkt zugeordnet ist. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` ist jedem Worker 1 DPU (4 vCPU, 16 GB Arbeitsspeicher, 64 GB Festplattenplatz) sowie ein Executor pro Worker zugewiesen. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge.
  - Für den Worker-Typ `G.2X` sind jedem Worker 2 DPU (8 vCPU, 32 GB Arbeitsspeicher, 128 GB Fewstplattenplatz) sowie ein Executor pro Worker zugewiesen. Wir empfehlen diesen Worker-Typ für speicherintensive Aufträge.

  Bekanntes Problem: Wenn ein Entwicklungsendpunkt mit der Konfiguration `G.2X` `WorkerType` erstellt wird, werden die Spark-Treiber für den Entwicklungsendpunkt auf 4 vCPU, 16 GB Arbeitsspeicher und einem 64 GB-Datenträger ausgeführt.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

Die Glue-Version bestimmt die Versionen von Apache Spark und Python, die AWS Glue unterstützt werden. Die Python-Version gibt die Version an, die für die Ausführung Ihrer ETL-Skripts auf Entwicklungsendpunkten unterstützt wird.

Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

Für Aufträge, die ohne Angabe einer Glue-Version erstellt werden, gilt standardmäßig Glue 0.9.

Sie können eine Version der Python-Unterstützung für Entwicklungsendpunkte angeben, indem Sie den `Arguments` Parameter im `CreateDevEndpoint` oder `UpdateDevEndpoint` APIs verwenden. Wenn keine Argumente angegeben werden, verwendet die Version standardmäßig Python 2.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die dem Entwicklungsendpunkt zugeordnet sind.

  Die maximale Anzahl von Workern, die Sie definieren können, ist 299 für `G.1X` und 149 für `G.2X`.

- `ExtraPythonLibsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Python-Bibliothek in einem Amazon S3-Bucket, die in Ihrem `DevEndpoint` geladen werden soll. Mehrere Werte müssen vollständige Pfade sein, die durch Kommata getrennt werden.

  > **Note**
  >
  > Sie können nur reine Python-Bibliotheken mit einem `DevEndpoint` verwenden. Bibliotheken, die auf C-Erweiterungen basieren, wie zum Beispiel die pandas-Python-Datenanalysebibliothek, werden derzeit nicht unterstützt.

- `ExtraJarsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Java-`.jar`-Datei in einem S3-Bucket, die in Ihren `DevEndpoint` geladen werden soll.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der `SecurityConfiguration`-Struktur, die mit diesem `DevEndpoint` verwendet werden soll.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die mit diesem DevEndpoint zu verwendenden Tags. Sie können Tags verwenden, um den Zugriff auf die DevEndpoint zu beschränken. Weitere Informationen zu Tags in AWS Glue finden Sie unter AWS Tags in AWS Glue im Entwicklerhandbuch.
- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Zuordnung von Argumenten, die zum Konfigurieren des `DevEndpoint`verwendet werden.

Antwort

- `EndpointName` – UTF-8-Zeichenfolge.

  Der dem neuen `DevEndpoint` zugewiesene Name.
- `Status` – UTF-8-Zeichenfolge.

  Der aktuelle Status des `DevEndpoint`.
- `SecurityGroupIds` – Ein UTF-8-Zeichenfolgen-Array.

  Die Sicherheitsgruppen, der dem neuen `DevEndpoint` zugewiesen sind.
- `SubnetId` – UTF-8-Zeichenfolge.

  Die Subnetz-ID des neuen `DevEndpoint`.
- `RoleArn` – UTF-8-Zeichenfolge, die AWS IAM ARN string pattern entspricht.

  Der Amazon-Ressourcenname (ARN) der Rolle des neuen `DevEndpoint`.
- `YarnEndpointAddress` – UTF-8-Zeichenfolge.

Die von diesem `DevEndpoint` verwendete Adresse des YARN-Endpunkts.

- `ZeppelinRemoteSparkInterpreterPort` – Zahl (Ganzzahl).

  Der Apache Zeppelin-Port für den Apache-Spark-Remote-Interpreter.

- `NumberOfNodes` – Zahl (Ganzzahl).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die diesem zugewiesen sind DevEndpoint.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der dem Entwicklungsendpunkt zugeordnet ist. Möglich sind die Werte Standard, G.1X oder G.2X.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Die Glue-Version bestimmt die Versionen von Apache Spark und Python, die AWS Glue unterstützt werden. Die Python-Version gibt die Version an, die für die Ausführung Ihrer ETL-Skripts auf Entwicklungsendpunkten unterstützt wird.

  Weitere Informationen zu den verfügbaren AWS Glue Versionen und den entsprechenden Spark- und Python-Versionen finden Sie unter Glue-Version im Entwicklerhandbuch.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType` die dem Entwicklungsendpunkt zugeordnet sind.

- `AvailabilityZone` – UTF-8-Zeichenfolge.

  Die AWS Availability Zone, in `DevEndpoint` der sich das befindet.

- `VpcId` – UTF-8-Zeichenfolge.

  Die ID der von diesem `DevEndpoint`verwendeten Virtual Private Cloud (VPC).

- `ExtraPythonLibsS3Path` – UTF-8-Zeichenfolge.

  Pfad(e) zu mindestens einer Python-Bibliothek in einem S3-Bucket, die in Ihrem `DevEndpoint` geladen werden wird.

- `ExtraJarsS3Path` – UTF-8-Zeichenfolge.

Pfad(e) zu mindestens einer Java-`.jar`-Datei in einem S3-Bucket, die in Ihren `DevEndpoint` geladen werden wird.

- `FailureReason` – UTF-8-Zeichenfolge.

  Der Grund für einen aktuellen Ausfall an diesem `DevEndpoint`.

- `SecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der `SecurityConfiguration`-Struktur, die mit diesem `DevEndpoint` verwendet werden soll.

- `CreatedTimestamp` – Zeitstempel.

  Der Zeitpunkt, an dem dieser `DevEndpoint` zuletzt geändert wurde.

- `Arguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die Zuordnung von Argumenten, die zum Konfigurieren dieses `DevEndpoint` verwendet werden.

  Gültige Argumente sind:

  - `"--enable-glue-datacatalog": ""`

  Sie können eine Version der Python-Unterstützung für Entwicklungsendpunkte angeben, indem Sie den `Arguments` Parameter im `CreateDevEndpoint` oder `UpdateDevEndpoint` APIs verwenden. Wenn keine Argumente angegeben werden, verwendet die Version standardmäßig Python 2.

Fehler

- `AccessDeniedException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`

- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`

UpdateDevEndpoint Aktion (Python: update_dev_endpoint)

Aktualisiert einen bestimmten Entwicklungsendpunkt.

Anforderung

- `EndpointName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name des zu aktualisierenden `DevEndpoint`.
- `PublicKey` – UTF-8-Zeichenfolge.

  Der öffentliche Schlüssel für den `DevEndpoint`.
- `AddPublicKeys` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 5 Zeichenfolgen.

  Die Liste öffentlicher Schlüssel für den zu verwendenden `DevEndpoint`.
- `DeletePublicKeys` – Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 5 Zeichenfolgen.

  Die Liste öffentlicher Schlüssel, die von dem `DevEndpoint`gelöscht werden sollen.
- `CustomLibraries` – Ein DevEndpointCustomLibraries-Objekt.

  Benutzerdefinierte Python- oder Java-Bibliotheken, die in den `DevEndpoint` geladen werden sollen.
- `UpdateEtlLibraries` – Boolesch.

  `True`, wenn die Liste der benutzerdefinierten Bibliotheken, die in den Entwicklungsendpunkt geladen werden sollen, aktualisiert werden muss, andernfalls `False`.
- `DeleteArguments` – Ein UTF-8-Zeichenfolgen-Array.

  Die Liste der Argumentenschlüssel, die aus der Zuordnung von Argumenten, die zum Konfigurieren des `DevEndpoint` verwendet werden, gelöscht werden sollen.
- `AddArguments` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 100 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge.

Jeder Wert ist eine UTF-8-Zeichenfolge.

Die Zuordnung von Argumenten, die zur Zuordnung von Argumenten, die zum Konfiguration des `DevEndpoint` verwendet werden, hinzugefügt werden sollen.

Gültige Argumente sind:

- `"--enable-glue-datacatalog": ""`

Sie können eine Version der Python-Unterstützung für Entwicklungsendpunkte angeben, indem Sie den `Arguments` Parameter im `CreateDevEndpoint` oder `UpdateDevEndpoint` APIs verwenden. Wenn keine Argumente angegeben werden, verwendet die Version standardmäßig Python 2.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`

DeleteDevEndpoint Aktion (Python: delete_dev_endpoint)

Löscht einen bestimmten Entwicklungsendpunkt.

Anforderung

- `EndpointName` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name der `DevEndpoint`.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoint Aktion (Python: get_dev_endpoint)

Ruft Informationen zu einem bestimmten Entwicklungsendpunkt ab.

> **Note**
>
> Beim Erstellen eines Entwicklungsendpunkts in einer Virtual Private Cloud (VPC) gibt AWS Glue nur eine private IP-Adresse zurück, und das Feld für die öffentliche IP-Adresse wird nicht ausgefüllt. Wenn Sie einen Nicht-VPC-Entwicklungsendpunkt erstellen, wird nur eine öffentliche IP-Adresse AWS Glue zurückgegeben.

Anforderung

- `EndpointName` – Erforderlich: UTF-8-Zeichenfolge.

  Name des `DevEndpoint`, für den Informationen abgerufen werden sollen.

Antwort

- `DevEndpoint` – Ein DevEndpoint-Objekt.

  Eine `DevEndpoint`-Definition.

Fehler

- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoints Aktion (Python: get_dev_endpoints)

Ruft alle Entwicklungsendpunkte in diesem Konto ab. AWS

> **Note**
>
> Beim Erstellen eines Entwicklungsendpunkts in einer Virtual Private Cloud (VPC) gibt AWS Glue nur eine private IP-Adresse zurück, und das Feld für die öffentliche IP-Adresse wird nicht ausgefüllt. Wenn Sie einen Nicht-VPC-Entwicklungsendpunkt erstellen, wird nur eine öffentliche IP-Adresse AWS Glue zurückgegeben.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Informationen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `DevEndpoints` – Ein Array mit DevEndpoint-Objekten.

  Eine Liste von `DevEndpoint`-Definitionen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungstoken, falls noch nicht alle `DevEndpoint`-Definitionen zurückgegeben wurden.

Fehler

- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

BatchGetDevEndpoints Aktion (Python: batch_get_dev_endpoints)

Gibt eine Liste von Ressourcen-Metadaten für eine bestimmte Liste von Entwicklungsendpunkt-Namen zurück. Nach dem Aufrufen der `ListDevEndpoints`-Operation können Sie diese Operation aufrufen, um auf die Daten zuzugreifen, für die Ihnen Berechtigungen erteilt wurden. Dieser Vorgang unterstützt alle IAM-Berechtigungen, einschließlich Berechtigungsbedingungen, die Tags verwenden.

Anforderung

- `customerAccountId` – UTF-8-Zeichenfolge.

  AWS Die Konto-ID.
- `DevEndpointNames` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste von `DevEndpoint`-Namen, die von der `ListDevEndpoint`-Operation als Namen zurückgegeben werden können.

Antwort

- `DevEndpoints` – Ein Array mit DevEndpoint-Objekten.

  Eine Liste von `DevEndpoint`-Definitionen.
- `DevEndpointsNotFound` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste der nicht gefunden `DevEndpoints`.

Fehler

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`

- `InvalidInputException`

ListDevEndpoints Aktion (Python: list_dev_endpoints)

Ruft die Namen aller `DevEndpoint`-Ressourcen in diesem AWS -Konto oder die Ressourcen mit dem angegebenen Tag ab. Mit dieser Operation können Sie sehen, welche Ressourcen in Ihrem Konto verfügbar sind, sowie deren Namen.

Diese Operation akzeptiert das optionale `Tags`-Feld, das Sie als Filter für die Antwort verwenden können, so dass markierte Ressourcen als Gruppe abgerufen werden können. Wenn Sie die Tag-Filterung verwenden, werden nur Ressourcen mit dem Tag abgerufen.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Gibt an, das nur diese markierten Ressourcen zurückgegeben werden sollen.

Antwort

- `DevEndpointNames` – Ein UTF-8-Zeichenfolgen-Array.

  Die Namen aller `DevEndpoint` im Konto oder der `DevEndpoint` mit den angegebenen Tags.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

Schema Registry

Die Schemaregistrierungs-API beschreibt die Datentypen und die API für die Arbeit mit Schemas in AWS Glue.

Datentypen

- RegistryId Struktur
- RegistryListItem Struktur
- MetadataInfo Struktur
- OtherMetadataValueListItem Struktur
- SchemaListItem Struktur
- SchemaVersionListItem Struktur
- MetadataKeyValuePair Struktur
- SchemaVersionErrorItem Struktur
- ErrorDetails Struktur
- SchemaVersionNumber Struktur
- SchemaId Struktur

RegistryId Struktur

Eine Wrapper-Struktur, die den Registrierungsnamen und den Amazon-Ressourcennamen (ARN) enthält.

Felder

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

Name der Registrierung. Wird nur für die Suche verwendet. Eine `RegistryArn` oder `RegistryName` muss angegeben werden.

- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  ARN der zu aktualisierenden Registrierung. Eine `RegistryArn` oder `RegistryName` muss angegeben werden.

RegistryListItem Struktur

Eine Struktur, die Details für eine Registrierung enthält.

Felder

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.

- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der Registrierung.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Registrierung.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `DELETING`).

  Der Status der Registrierung.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Das Datum, an dem die Registrierung erstellt wurde.

- `UpdatedTime` – UTF-8-Zeichenfolge.

  Das Datum, an dem die Registrierung aktualisiert wurde.

MetadataInfo Struktur

Eine Struktur, die Metadateninformationen für eine Schemaversion enthält.

Felder

- `MetadataValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #14.

  Der entsprechende Wert eines Metadatenschlüssels.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Die Uhrzeit, zu der der Eintrag erstellt wurde.

- `OtherMetadataValueList` – Ein Array mit OtherMetadataValueListItem-Objekten.

  Andere Metadaten, die zum gleichen Metadatenschlüssel gehören.

OtherMetadataValueListItem Struktur

Eine Struktur, die andere Metadaten für eine Schemaversion enthält, die zu demselben Metadatenschlüssel gehört.

Felder

- `MetadataValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #14.

  Der entsprechende Wert des Metadatenschlüssels für die anderen Metadaten, die zum selben Metadatenschlüssel gehören.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Die Uhrzeit, zu der der Eintrag erstellt wurde.

SchemaListItem Struktur

Ein Objekt, das minimale Details für ein Schema enthält.

Felder

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung, in der sich das Schema befindet.

- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) für das Schema.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung für das Schema.

- `SchemaStatus` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `DELETING`).

  Der Status des Schemas.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Datum und Uhrzeit, an denen das Schema angelegt wurde.

- `UpdatedTime` – UTF-8-Zeichenfolge.

  Datum und Uhrzeit, an denen das Schema aktualisiert wurde.

SchemaVersionListItem Struktur

Ein Objekt, das die Details zu einer Schemaversion enthält.

Felder

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Kennung der Schemaversion.
- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Der Status der Schemaversion.
- `CreatedTime` – UTF-8-Zeichenfolge.

  Das Datum und die Uhrzeit, zu denen die Schemaversion erstellt wurde.

MetadataKeyValuePair Struktur

Eine Struktur, die Schlüssel-Wert-Paare für Metadaten enthält.

Felder

- `MetadataKey` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #14.

  Ein Metadatenschlüssel.
- `MetadataValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #14.

  Der entsprechende Wert eines Metadatenschlüssels.

SchemaVersionErrorItem Struktur

Ein Objekt, das die Fehlerdetails für einen Vorgang in einer Schemaversion enthält.

Felder

- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.
- `ErrorDetails` – Ein ErrorDetails-Objekt.

Die Fehlerdetails für die Schemaversion.

ErrorDetails Struktur

Ein Objekt, das Fehlerdetails enthält.

Felder

- `ErrorCode` – UTF-8-Zeichenfolge.

  Der Fehlercode für einen Fehler.
- `ErrorMessage` – UTF-8-Zeichenfolge.

  Die Fehlermeldung für einen Fehler.

SchemaVersionNumber Struktur

Eine Struktur, die Informationen für eine Schemaversion enthält.

Felder

- `LatestVersion` – Boolesch.

  Die neueste Version, die für das Schema verfügbar ist.
- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

SchemaId Struktur

Die eindeutige ID des Schemas in der AWS Glue Schemaregistrierung.

Felder

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.

- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Schema Registry, die das Schema enthält

Operationen

- CreateRegistry Aktion (Python: create_registry)
- CreateSchema Aktion (Python: create_schema)
- GetSchema Aktion (Python: get_schema)
- ListSchemaVersions Aktion (Python: list_schema_versions)
- GetSchemaVersion Aktion (Python: get_schema_version)
- GetSchemaVersionsDiff Aktion (Python: get_schema_versions_diff)
- ListRegistries Aktion (Python: list_registries)
- ListSchemas Aktion (Python: list_schemas)
- RegisterSchemaVersion Aktion (Python: register_schema_version)
- UpdateSchema Aktion (Python: update_schema)
- CheckSchemaVersionValidity Aktion (Python: check_schema_version_validity)
- UpdateRegistry Aktion (Python: update_registry)
- GetSchemaByDefinition Aktion (Python: get_schema_by_definition)
- GetRegistry Aktion (Python: get_registry)
- PutSchemaVersionMetadata Aktion (Python: put_schema_version_metadata)
- QuerySchemaVersionMetadata Aktion (Python: query_schema_version_metadata)
- RemoveSchemaVersionMetadata Aktion (Python: remove_schema_version_metadata)
- DeleteRegistry Aktion (Python: delete_registry)
- DeleteSchema Aktion (Python: delete_schema)
- DeleteSchemaVersions Aktion (Python: delete_schema_versions)

CreateRegistry Aktion (Python: create_registry)

Erstellt eine neue Registrierung, mit der eine Schemasammlung gespeichert werden kann.

Anforderung

- `RegistryName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der zu erstellenden Registrierung. Darf höchstens 255 Zeichen enthalten und nur aus Buchstaben, Ziffern, Bindestrichen, Unterstrichen, Dollarzeichen und Hash-Zeichen bestehen. Keine Leerzeichen.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Registrierung. Wenn keine Beschreibung angegeben wird, gibt es dafür keinen Standardwert.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  AWS Tags, die ein Schlüssel-Wert-Paar enthalten und über die Konsole, Befehlszeile oder API durchsucht werden können.

Antwort

- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der neu erstellten Registrierung.
- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

Eine Beschreibung der Registrierung.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags für die Registrierung.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

CreateSchema Aktion (Python: create_schema)

Erstellt einen neuen Schemasatz und registriert die Schemadefinition. Gibt einen Fehler zurück, wenn der Schemasatz bereits vorhanden ist, ohne die Version tatsächlich zu registrieren.

Wenn der Schemasatz erstellt wird, wird ein Versionsprüfpunkt auf die erste Version gesetzt. Der Kompatibilitätsmodus „DISABLED“ (DEAKTIVIERT) schränkt das Hinzufügen zusätzlicher Schemaversionen nach der ersten Schemaversion ein. Für alle anderen Kompatibilitätsmodi wird die Validierung von Kompatibilitätseinstellungen erst ab der zweiten Version angewendet, wenn die `RegisterSchemaVersion` API verwendet wird.

Wird diese API ohne `RegistryId` aufgerufen, wird ein Eintrag für eine „default-registry“ (Standardregistrierung) in den Registry-Datenbanktabellen erstellt, sofern nicht bereits vorhanden.

Anforderung

- `RegistryId` – Ein RegistryId-Objekt.

Das ist ein Wrapper-Shape, das die Registry-Identitätsfelder enthält. Falls dieses nicht angegeben wird, wird die Standardregistrierung verwendet. Das ARN-Format dafür lautet: `arn:aws:glue:us-east-2:<customer id>:registry/default-registry:random-5-letter-id`.

- `SchemaName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des zu erstellenden Schemas darf höchstens 255 Zeichen enthalten und nur aus Buchstaben, Ziffern, Bindestrichen, Unterstrichen, Dollarzeichen und Hash-Zeichen bestehen. Keine Leerzeichen.

- `DataFormat` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.

- `Compatibility` – UTF-8-Zeichenfolge (zulässige Werte: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Der Kompatibilitätsmodus des Schemas. Die möglichen Werte sind:

  - NONE (KEINE): Es gibt keinen Kompatibilitätsmodus. Sie können diese Option in Entwicklungsszenarien verwenden, oder wenn Sie den Kompatibilitätsmodus, den Sie auf Schemata anwenden möchten, nicht kennen. Jede neue Version, die hinzugefügt wird, wird ohne Kompatibilitätsprüfung akzeptiert.
  - DISABLED (DEAKTIVIERT): Diese Kompatibilitätsoption verhindert das Versioning für ein bestimmtes Schema. Sie können diese Option verwenden, um zukünftiges Versioning eines Schemas zu verhindern.
  - BACKWARD (ABWÄRTS): Diese Kompatibilitätsoption wird empfohlen, damit Daten-Receiver sowohl die aktuelle als auch die vorherige Schemaversion lesen können. Das bedeutet, dass beispielsweise eine neue Schemaversion Datenfelder nicht löschen oder den Typ dieser Felder ändern kann, sodass sie von Readern mit der vorherigen Version nicht gelesen werden können.
  - BACKWARD_ALL (ABWÄRTS_ALLE): Diese Kompatibilitätsoption ermöglicht den Daten-Receivern sowohl die aktuelle als auch alle vorherigen Schemaversionen zu lesen. Sie können diese Option verwenden, wenn Sie Felder löschen oder optionale Felder hinzufügen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.
  - FORWARD (AUFWÄRTS): Diese Kompatibilitätsoption ermöglicht Daten-Receivern, sowohl die aktuelle als auch die nachfolgende Schemaversion zu lesen, aber nicht unbedingt spätere Versionen. Sie können diese Option verwenden, wenn Sie Felder hinzufügen oder optionale

Felder löschen und nur die Kompatibilität mit der letzten vorherigen Schemaversion überprüfen möchten.

- FORWARD_ALL (AUFWÄRTS_ALLE): Diese Kompatibilitätsoption ermöglicht Daten-Receivern, Daten zu lesen, die von Verfassern eines neuen registrierten Schemas geschrieben wurden. Sie können diese Option verwenden, wenn Sie Felder hinzufügen oder optionale Felder löschen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.
- FULL (VOLL): Diese Kompatibilitätsoption ermöglicht Data-Receivern, Daten zu lesen, die von Verfassern geschrieben wurden, die die vorherige oder die nächste Version des Schemas verwenden, jedoch nicht unbedingt frühere oder spätere Versionen. Sie können diese Option verwenden, wenn Sie optionale Felder hinzufügen oder löschen und nur die Kompatibilität mit der letzten vorherigen Schemaversion überprüfen möchten.
- FULL_ALL (VOLL_ALLE): Diese Kompatibilitätsoption ermöglicht Daten-Receiver, Daten zu lesen, die von Verfassern geschrieben wurden, die alle früheren Schemaversionen verwenden. Sie können diese Option verwenden, wenn Sie optionale Felder hinzufügen oder löschen und die Kompatibilität mit allen vorherigen Schemaversionen überprüfen möchten.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine optionale Beschreibung des Schemas. Wenn keine Beschreibung angegeben wird, gibt es dafür keinen automatischen Standardwert.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  AWS Tags, die ein Schlüssel-Wert-Paar enthalten und über die Konsole, Befehlszeile oder API durchsucht werden können. Falls angegeben, folgt es dem AWS tags-on-create Muster.

- `SchemaDefinition` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 170 000 Bytes lang, passend zum Custom string pattern #13.

  Die Schemadefinition, die die `DataFormat`-Einstellung für `SchemaName` verwendet.

Antwort

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.

- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der Registrierung.

- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Schemas, das gegebenenfalls bei der Erstellung angegeben wurde.

- `DataFormat` – UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.

- `Compatibility` – UTF-8-Zeichenfolge (zulässige Werte: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Der Schema-Kompatibilitätsmodus.

- `SchemaCheckpoint` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Checkpoints (das letzte Mal, als der Kompatibilitätsmodus geändert wurde).

- `LatestSchemaVersion` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die aktuellste Version des Schemas, das der zurückgegebenen Schemadefinition zugewiesen ist.

- `NextSchemaVersion` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

Die nächste Version des Schemas, das der zurückgegebenen Schemadefinition zugewiesen ist.

- `SchemaStatus` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `DELETING`).

  Der Status des Schemas.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags für das Schema.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Kennung der ersten Schemaversion.

- `SchemaVersionStatus` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Der Status der ersten erstellten Schemaversion.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

GetSchema Aktion (Python: get_schema)

Beschreibt das angegebene Schema im Detail.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
  - SchemaId$SchemaName: Der Name des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.

Antwort

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.

- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der Registrierung.

- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Schemas, das gegebenenfalls bei der Erstellung angegeben wurde

- `DataFormat` – UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.

- `Compatibility` – UTF-8-Zeichenfolge (zulässige Werte: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

Der Kompatibilitätsmodus des Schemas.

- `SchemaCheckpoint` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Checkpoints (das letzte Mal, als der Kompatibilitätsmodus geändert wurde).
- `LatestSchemaVersion` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die aktuellste Version des Schemas, das der zurückgegebenen Schemadefinition zugewiesen ist.
- `NextSchemaVersion` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die nächste Version des Schemas, das der zurückgegebenen Schemadefinition zugewiesen ist.
- `SchemaStatus` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `DELETING`).

  Der Status des Schemas.
- `CreatedTime` – UTF-8-Zeichenfolge.

  Das Datum und die Uhrzeit, zu denen das Schema erstellt wurde.
- `UpdatedTime` – UTF-8-Zeichenfolge.

  Datum und Uhrzeit, an denen das Schema aktualisiert wurde.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

ListSchemaVersions Aktion (Python: list_schema_versions)

Gibt eine Liste der von Ihnen erstellten Schemaversionen mit minimalen Informationen zurück. Der Status „Deleted“ (Gelöscht) wird nicht in die Ergebnisse aufgenommen. Leere Ergebnisse werden zurückgegeben, wenn keine Schemaversionen verfügbar sind.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

- SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
- SchemaId$SchemaName: Der Name des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Maximale Anzahl der erforderlichen Ergebnisse pro Seite. Wenn der Wert nicht angegeben wird, wird der Standardwert auf 25 pro Seite gesetzt.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `Schemas` – Ein Array mit SchemaVersionListItem-Objekten.

  Ein Array von `SchemaVersionList`-Objekten, die Details jeder Schemaversion enthalten.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

GetSchemaVersion Aktion (Python: get_schema_version)

Ruft das angegebene Schema anhand seiner eindeutigen ID ab, die zugewiesen wird, wenn eine Version des Schemas erstellt oder registriert wird. Der Status „Deleted“ (Gelöscht) wird nicht in die Ergebnisse aufgenommen.

Anforderung

- `SchemaId` – Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
  - SchemaId$SchemaName: Der Name des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die `SchemaVersionId` der Schemaversion. Dieses Feld ist für das Abrufen nach Schema-ID erforderlich. Entweder dies oder der `SchemaId`-Wrapper muss zur Verfügung gestellt werden.
- `SchemaVersionNumber` – Ein SchemaVersionNumber-Objekt.

  Die Versionsnummer des Schemas.

Antwort

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die `SchemaVersionId` der Schemaversion.
- `SchemaDefinition` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 170 000 Bytes lang, passend zum Custom string pattern #13.

  Die Schemadefinition für die Schema-ID.
- `DataFormat` – UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.
- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.
- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Der Status der Schemaversion.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Das Datum und die Uhrzeit, zu denen die Schemaversion erstellt wurde.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

GetSchemaVersionsDiff Aktion (Python: get_schema_versions_diff)

Ruft die Abweichung der Schemaversion im angegebenen Differenztyp zwischen zwei gespeicherten Schemaversionen in der Schema Registry ab.

Mit dieser API können Sie zwei Schemaversionen zwischen zwei Schemadefinitionen unter demselben Schema vergleichen.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.
  - SchemaId$SchemaName: Der Name des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.

- `FirstSchemaVersionNumber` – Erforderlich: Ein SchemaVersionNumber-Objekt.

  Die erste der beiden zu vergleichenden Schemaversionen.

- `SecondSchemaVersionNumber` – Erforderlich: Ein SchemaVersionNumber-Objekt.

  Die zweite der beiden zu vergleichenden Schemaversionen.

- `SchemaDiffType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `SYNTAX_DIFF`).

  Bezieht sich auf `SYNTAX_DIFF`, bei dem es sich um den derzeit unterstützten Differenztyp handelt.

Antwort

- `Diff` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 340 000 Bytes lang, passend zum Custom string pattern #13.

  Der Unterschied zwischen Schemas als Zeichenfolge im JsonPatch Format.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`

ListRegistries Aktion (Python: list_registries)

Gibt eine Liste der von Ihnen erstellten Registrierungen mit minimalen Registrierungsinformationen zurück. Registrierungen im `Deleting`-Status werden nicht in die Ergebnisse einbezogen. Leere Ergebnisse werden zurückgegeben, wenn keine Registrierungen verfügbar sind.

Anforderung

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Maximale Anzahl der erforderlichen Ergebnisse pro Seite. Wenn der Wert nicht angegeben wird, wird der Standardwert auf 25 pro Seite gesetzt.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `Registries` – Ein Array mit RegistryListItem-Objekten.

Ein Array von `RegistryDetailedListItem`-Objekten, die minimale Details jeder Registrierung enthalten.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

ListSchemas Aktion (Python: list_schemas)

Gibt eine Liste von Schemas mit minimalen Details zurück. Schemas mit dem Status „Deleting“ (Wird gelöscht) werden nicht in die Ergebnisse aufgenommen. Leere Ergebnisse werden zurückgegeben, wenn keine Schemas verfügbar sind.

Wenn die `RegistryId` nicht angegeben wird, sind alle Schemas in den Registry-Listen Teil der API-Antwort.

Anforderung

- `RegistryId` – Ein RegistryId-Objekt.

  Eine Wrapper-Struktur, die den Registrierungsnamen und den Amazon-Ressourcennamen (ARN) enthält.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 100.

  Maximale Anzahl der erforderlichen Ergebnisse pro Seite. Wenn der Wert nicht angegeben wird, wird der Standardwert auf 25 pro Seite gesetzt.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `Schemas` – Ein Array mit SchemaListItem-Objekten.

  Ein Array von `SchemaListItem`-Objekten, die Details jedes Schemas enthalten.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

RegisterSchemaVersion Aktion (Python: register_schema_version)

Fügt zum vorhandenen Schema eine neue Version hinzu. Gibt einen Fehler zurück, wenn die neue Version des Schemas die Kompatibilitätsanforderungen des Schemasatzes nicht erfüllt. Diese API erstellt keinen neuen Schemasatz und gibt einen 404-Fehler zurück, wenn der Schemasatz nicht bereits in der Schema Registry vorhanden ist.

Handelt es sich hierbei um die erste Schemadefinition, die in der Schema Registry registriert wird, speichert diese API die Schemaversion und gibt sie sofort zurück. Andernfalls kann dieser Aufruf aufgrund von Kompatibilitätsmodi länger als andere Vorgänge ausgeführt werden. Rufen Sie die `GetSchemaVersion`-API mit der `SchemaVersionId` auf, um die Kompatibilitätsmodi zu überprüfen.

Wenn dieselbe Schemadefinition bereits in der Schema Registry als Version gespeichert ist, wird die Schema-ID des vorhandenen Schemas zurückgegeben.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
  - SchemaId$SchemaName: Der Name des Schemas. Entweder `SchemaArn` oder `SchemaName` und `RegistryName` müssen zur Verfügung gestellt werden.
- `SchemaDefinition` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 170 000 Bytes lang, passend zum Custom string pattern #13.

  Die Schemadefinition, die die `DataFormat`-Einstellung für `SchemaName` verwendet.

Antwort

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige ID, die die Version dieses Schemas darstellt.
- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Version dieses Schemas (nur für den Synchronisierungsfluss, falls es sich um die erste Version handelt).
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Der Status der Schemaversion.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

UpdateSchema Aktion (Python: update_schema)

Aktualisiert die Beschreibung, die Kompatibilitätseinstellung oder den Versionsprüfpunkt für einen Schemasatz.

Beim Aktualisieren der Kompatibilitätseinstellung überprüft der Aufruf nicht die Kompatibilität für den gesamten Satz an Schemaversionen mit der neuen Kompatibilitätseinstellung. Wird der Wert für `Compatibility` bereitgestellt, wird die `VersionNumber` (ein Prüfpunkt) ebenfalls benötigt. Die API überprüft die Versionsnummer des Prüfpunkts aus Konsistenzgründen.

Wenn der Wert für die `VersionNumber` (Prüfpunkt) zur Verfügung gestellt wird, ist `Compatibility` optional und kann verwendet werden, um einen Prüfpunkt für das Schema festzulegen/zurückzusetzen.

Dieses Update wird nur durchgeführt, wenn sich das Schema im Status AVAILABLE (VERFÜGBAR) befindet.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:

  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.
  - SchemaId$SchemaName: Der Name des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.
- `SchemaVersionNumber` – Ein SchemaVersionNumber-Objekt.

  Versionsnummer für Prüfpunkt erforderlich. Eine `VersionNumber` oder `Compatibility` muss angegeben werden.
- `Compatibility` – UTF-8-Zeichenfolge (zulässige Werte: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Die neue Kompatibilitätseinstellung für das Schema.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die neue Beschreibung für das Schema.

Antwort

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.
- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.
- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung, die das Schema enthält

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InternalServiceException`

CheckSchemaVersionValidity Aktion (Python: check_schema_version_validity)

Validiert das bereitgestellte Schema. Dieser Aufruf hat keine Nebeneffekte, er validiert einfach nur mit dem bereitgestellten Schema und `DataFormat` als Format. Da kein Name eines Schemasatzes verwendet wird, werden keine Kompatibilitätsprüfungen durchgeführt.

Anforderung

- `DataFormat` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.
- `SchemaDefinition` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 170 000 Bytes lang, passend zum Custom string pattern #13.

Die Definition des Schemas, das validiert werden muss.

Antwort

- `Valid` – Boolesch.

  Gibt true zurück, wenn das Schema gültig ist, andernfalls false.

- `Error` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 5 000 Bytes lang.

  Eine Fehlermeldung für einen Validierungsfehler.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

UpdateRegistry Aktion (Python: update_registry)

Aktualisiert eine vorhandene Registrierung, die zum Speichern einer Sammlung von Schemas verwendet wird. Die aktualisierten Eigenschaften beziehen sich auf die Registrierung und ändern keines der Schemas in der Registrierung.

Anforderung

- `RegistryId` – Erforderlich: Ein RegistryId-Objekt.

  Das ist ein Wrapper-Struktur, die den Registrierungsnamen und den Amazon-Ressourcennamen (ARN) enthält.

- `Description` – Erforderlich: Beschreibende Zeichenfolge, nicht mehr als 2 048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Registrierung. Bei keiner Angabe einer Beschreibung wird dieses Feld nicht aktualisiert.

Antwort

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der aktualisierten Registrierung.
- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der aktualisierten Registrierung.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InternalServiceException`

GetSchemaByDefinition Aktion (Python: get_schema_by_definition)

Ruft ein Schema von der `SchemaDefinition` ab. Die Schemadefinition wird an die Schema Registry gesendet, kanonisiert und gehasht. Wenn der Hash innerhalb des Bereichs des `SchemaName` oder ARN (oder der Standardregistrierung, falls keine angegeben ist) abgeglichen wird, werden die Metadaten dieses Schemas zurückgegeben. Andernfalls wird ein 404-Fehler oder ein Fehler zurückgegeben. NotFound Schemaversionen mit dem Status `Deleted` werden nicht in die Ergebnisse aufgenommen.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die Schema-Identitätsfelder enthält. Die Struktur enthält:
  - SchemaId$SchemaArn: Der Amazon-Ressourcenname (ARN) des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.
  - SchemaId$SchemaName: Der Name des Schemas. Eine `SchemaArn` oder `SchemaName` muss angegeben werden.

- `SchemaDefinition` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 170 000 Bytes lang, passend zum Custom string pattern #13.

  Die Definition des Schemas, für das Schemadetails erforderlich sind.

Antwort

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die Schema-ID der Schemaversion.

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.

- `DataFormat` – UTF-8-Zeichenfolge (zulässige Werte: `AVRO` | `JSON` | `PROTOBUF`).

  Das Datenformat der Schemadefinition. Derzeit werden `AVRO`, `JSON` und `PROTOBUF` unterstützt.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Der Status der Schemaversion.

- `CreatedTime` – UTF-8-Zeichenfolge.

  Das Datum und die Uhrzeit, zu denen das Schema erstellt wurde.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

GetRegistry Aktion (Python: get_registry)

Beschreibt die angegebene Registrierung im Detail.

Anforderung

- `RegistryId` – Erforderlich: Ein RegistryId-Objekt.

  Das ist ein Wrapper-Struktur, die den Registrierungsnamen und den Amazon-Ressourcennamen (ARN) enthält.

Antwort

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.
- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der Registrierung.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Registrierung.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `DELETING`).

  Der Status der Registrierung.
- `CreatedTime` – UTF-8-Zeichenfolge.

  Datum und Uhrzeit der Erstellung der Registrierung.
- `UpdatedTime` – UTF-8-Zeichenfolge.

  Datum und Uhrzeit der Aktualisierung der Registrierung.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

PutSchemaVersionMetadata Aktion (Python: put_schema_version_metadata)

Legt das Metadaten-Schlüssel-Wert-Paar für eine angegebene Schemaversions-ID fest. Pro Schemaversion sind maximal 10 Schlüssel-Wert-Paare zulässig. Sie können über einen oder mehrere Aufrufe hinzugefügt werden.

Anforderung

- `SchemaId` – Ein SchemaId-Objekt.

  Die eindeutige ID für das Schema.
- `SchemaVersionNumber` – Ein SchemaVersionNumber-Objekt.

  Die Versionsnummer des Schemas.
- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Versions-ID der Schemaversion.
- `MetadataKeyValue` – Erforderlich: Ein MetadataKeyValuePair-Objekt.

  Der entsprechende Wert eines Metadatenschlüssels.

Antwort

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) für das Schema.
- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.
- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name für die Registrierung.
- `LatestVersion` – Boolesch.

Die neueste Version des Schemas.

- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Versions-ID der Schemaversion.

- `MetadataKey` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #14.

  Der Metadatenschlüssel.

- `MetadataValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #14.

  Der Wert des Metadatenschlüssels.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`

QuerySchemaVersionMetadata Aktion (Python: query_schema_version_metadata)

Fragt die Metadateninformationen der Schemaversion ab.

Anforderung

- `SchemaId` – Ein SchemaId-Objekt.

  Eine Wrapper-Struktur, die den Schemanamen und den Amazon-Ressourcennamen (ARN) enthält.

- `SchemaVersionNumber` – Ein SchemaVersionNumber-Objekt.

Die Versionsnummer des Schemas.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Versions-ID der Schemaversion.

- `MetadataList` – Ein Array mit MetadataKeyValuePair-Objekten.

  Durchsucht Schlüssel-Wert-Paare nach Metadaten, wenn sie nicht bereitgestellt werden, werden alle Metadateninformationen abgerufen.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 50.

  Maximale Anzahl der erforderlichen Ergebnisse pro Seite. Wenn der Wert nicht angegeben wird, wird der Standardwert auf 25 pro Seite gesetzt.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies ein Fortsetzungsaufruf ist.

Antwort

- `MetadataInfoMap` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, und entspricht dem Custom string pattern #14.

  Jeder Wert ist ein A MetadataInfo-Objekt.

  Die Zuordnung eines Metadatenschlüssels und der zugehörigen Werte.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Versions-ID der Schemaversion.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token zum Paginieren der zurückgegebenen Token-Liste. Wird zurückgegeben, wenn das aktuelle Segment der Liste nicht das letzte ist.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`

RemoveSchemaVersionMetadata Aktion (Python: remove_schema_version_metadata)

Entfernt ein Schlüssel-Wert-Paar aus den Metadaten der Schemaversion für die angegebene Schemaversions-ID.

Anforderung

- `SchemaId` – Ein SchemaId-Objekt.

  Eine Wrapper-Struktur, die den Schemanamen und den Amazon-Ressourcennamen (ARN) enthält.
- `SchemaVersionNumber` – Ein SchemaVersionNumber-Objekt.

  Die Versionsnummer des Schemas.
- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die eindeutige Versions-ID der Schemaversion.
- `MetadataKeyValue` – Erforderlich: Ein MetadataKeyValuePair-Objekt.

  Der Wert des Metadatenschlüssels.

Antwort

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des Schemas.
- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas.

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung.

- `LatestVersion` – Boolesch.

  Die neueste Version des Schemas.

- `VersionNumber` – Zahl (lang), nicht kleiner als 1 oder größer als 100 000.

  Die Versionsnummer des Schemas.

- `SchemaVersionId` – UTF-8-Zeichenfolge, nicht weniger als 36 oder mehr als 36 Bytes lang, passend zum Custom string pattern #44.

  Die Versions-ID für die Schemaversion.

- `MetadataKey` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #14.

  Der Metadatenschlüssel.

- `MetadataValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 256 Bytes lang, passend zum Custom string pattern #14.

  Der Wert des Metadatenschlüssels.

Fehler

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`

DeleteRegistry Aktion (Python: delete_registry)

Löscht die gesamte Registrierung einschließlich Schema und aller Versionen. Um den Status des Löschvorgangs zu erhalten, können Sie die `GetRegistry`-API nach dem asynchronen Aufruf aufrufen. Durch das Löschen einer Registrierung werden alle Online-Operationen für die Registrierung deaktiviert, z. B. das`UpdateRegistry`,`CreateSchema`, `UpdateSchema` und. `RegisterSchemaVersion` APIs

Anforderung

- `RegistryId` – Erforderlich: Ein RegistryId-Objekt.

  Das ist ein Wrapper-Struktur, die den Registrierungsnamen und den Amazon-Ressourcennamen (ARN) enthält.

Antwort

- `RegistryName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name der Registrierung, die gelöscht wird.
- `RegistryArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) der zu löschenden Registrierung.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `DELETING`).

  Der Status der Registrierung. Ein erfolgreicher Vorgang gibt den Status `Deleting` zurück.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

DeleteSchema Aktion (Python: delete_schema)

Löscht den gesamten Schemasatz, einschließlich des Schemasatzes und aller zugehörigen Versionen. Um den Status des Löschvorgangs zu erhalten, können Sie `GetSchema`-API nach dem asynchronen Aufruf aufrufen. Durch das Löschen einer Registrierung werden alle Online-Operationen für das Schema deaktiviert, z. B. das`GetSchemaByDefinition`, und. `RegisterSchemaVersion` APIs

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die den Schemanamen und den Amazon-Ressourcennamen (ARN) enthält.

Antwort

- `SchemaArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) des zu löschenden Schemas.
- `SchemaName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #45.

  Der Name des Schemas, das gelöscht wird.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `AVAILABLE` | `PENDING` | `DELETING`).

  Der Status des Schemas.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

DeleteSchemaVersions Aktion (Python: delete_schema_versions)

Entfernt Versionen aus dem angegebenen Schema. Eine Versionsnummer oder ein Bereich kann angegeben werden. Wenn der Kompatibilitätsmodus das Löschen einer benötigten Version wie BACKWARDS_FULL (ABWÄRTS_VOLL) verbietet, wird ein Fehler zurückgegeben. Das Aufrufen der `GetSchemaVersions`-API nach diesem Aufruf listet den Status der gelöschten Versionen auf.

Wenn der Bereich der Versionsnummern eine Version mit Prüfpunkt enthält, gibt die API einen 409-Konflikt zurück und fährt nicht mit dem Löschen fort. Sie müssen den Prüfpunkt zuerst mit der `DeleteSchemaCheckpoint`-API entfernen, bevor Sie diese API verwenden können.

Sie können mit der `DeleteSchemaVersions`-API nicht die erste Schemaversion im Schemasatz löschen. Die erste Schemaversion kann nur durch die `DeleteSchema`-API gelöscht werden. Dieser Vorgang löscht auch die unter den Schemaversionen angehängten `SchemaVersionMetadata`. Feste Löschungen werden für die Datenbank erzwungen.

Wenn der Kompatibilitätsmodus das Löschen einer benötigten Version wie BACKWARDS_FULL (ABWÄRTS_VOLL) verbietet, wird ein Fehler zurückgegeben.

Anforderung

- `SchemaId` – Erforderlich: Ein SchemaId-Objekt.

  Das ist eine Wrapper-Struktur, die den Schemanamen und den Amazon-Ressourcennamen (ARN) enthält.
- `Versions` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 100 000 Bytes lang, passend zum Custom string pattern #15.

  Es kann ein Versionsbereich im folgenden Format angegeben werden:
  - eine einzelne Versionsnummer, 5
  - ein Bereich, 5-8: löscht die Versionen 5, 6, 7, 8

Antwort

- `SchemaVersionErrors` – Ein Array mit SchemaVersionErrorItem-Objekten.

  Eine Liste von `SchemaVersionErrorItem`-Objekten, die jeweils eine Fehler- und Schemaversion enthalten.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

Workflows

Die Workflow-API beschreibt die Datentypen und die API im Zusammenhang mit der Erstellung, Aktualisierung oder Anzeige von Workflows in AWS Glue. Der Verlauf der Auftragsausführung ist 90 Tage lang für Ihren Workflow und Ihre Auftragsausführung zugänglich.

Datentypen

- JobNodeDetails Struktur
- CrawlerNodeDetails Struktur
- TriggerNodeDetails Struktur
- Crawl-Struktur
- Knotenstruktur
- Edge-Struktur
- Workflow-Struktur
- WorkflowGraph Struktur
- WorkflowRun Struktur
- WorkflowRunStatistics Struktur
- StartingEventBatchCondition Struktur
- Blueprint-Struktur
- BlueprintDetails Struktur
- LastActiveDefinition Struktur
- BlueprintRun Struktur

JobNodeDetails Struktur

Die Details einer Auftragsknotendarstellung im Workflow.

Felder

- `JobRuns` – Ein Array mit JobRun-Objekten.

  Die durch den Auftragsknoten repräsentierten Informationen für die Auftragsausführungen.

CrawlerNodeDetails Struktur

Die Details einer Crawler-Knotendarstellung im Workflow.

Felder

- `Crawls` – Ein Array mit Crawl-Objekten.

  Eine Liste von Crawlern, die durch den Crawler-Knoten repräsentiert werden.

TriggerNodeDetails Struktur

Die Details einer Auslöser-Knotendarstellung im Workflow.

Felder

- `Trigger` – Ein Auslöser-Objekt.

  Die Informationen des Auslösers, der durch den Auslöserknoten repräsentiert wird.

Crawl-Struktur

Die Details eines Crawl im Workflow.

Felder

- `State` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  Der Status des Crawlers.
- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Crawl gestartet wurde.
- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Crawl abgeschlossen wurde.
- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die dem Crawl zugeordnete Fehlermeldung.

- `LogGroup` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log group string pattern.

  Die dem Crawl zugeordnete Protokollgruppe.

- `LogStream` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang, passend zum Log-stream string pattern.

  Der dem Crawl zugeordnete Protokoll-Stream.

Knotenstruktur

Ein Knoten steht für eine AWS Glue Komponente (Trigger, Crawler oder Job) in einem Workflow-Diagramm.

Felder

- `Type` – UTF-8-Zeichenfolge (zulässige Werte: `CRAWLER` | `JOB` | `TRIGGER`).

  Der AWS Glue Komponententyp, der durch den Knoten repräsentiert wird.

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der AWS Glue Komponente, die durch den Knoten repräsentiert wird.

- `UniqueId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige ID, die dem Knoten innerhalb des Workflows zugewiesen ist.

- `TriggerDetails` – Ein TriggerNodeDetails-Objekt.

  Nähere Informationen zum Auslöser, wenn der Knoten einen Auslöser repräsentiert.

- `JobDetails` – Ein JobNodeDetails-Objekt.

  Nähere Informationen zum Auftrag, wenn der Knoten einen Auftrag repräsentiert.

- `CrawlerDetails` – Ein CrawlerNodeDetails-Objekt.

  Nähere Informationen zum Cawler, wenn der Knoten einen Crawler repräsentiert.

Edge-Struktur

Eine Kante steht für eine gerichtete Verbindung zwischen zwei AWS Glue Komponenten, die Teil des Workflows sind, zu dem die Kante gehört.

Felder

- `SourceId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige ID des Knotens innerhalb des Workflows, an dem die Edge beginnt.
- `DestinationId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige ID des Knotens innerhalb des Workflows, an dem die Edge endet.

Workflow-Struktur

Ein Workflow ist eine Sammlung mehrerer abhängiger AWS Glue Jobs und Crawler, die ausgeführt werden, um eine komplexe ETL-Aufgabe abzuschließen. Ein Workflow verwaltet die Ausführung und Überwachung aller zugehöriger Aufträge und Crawler.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows.
- `Description` – UTF-8-Zeichenfolge.

  Eine Beschreibung des Workflows.
- `DefaultRunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

Eine Sammlung von Eigenschaften, die als Teil einer jeden Ausführung des Workflows verwendet werden sollen. Die Ausführungseigenschaften werden jedem Auftrag im Workflow zur Verfügung gestellt. Ein Auftrag kann die Eigenschaften für die nächsten Aufträge im Workflow ändern.

- `CreatedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Workflow erstellt wurde.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Workflow zuletzt geändert wurde.

- `LastRun` – Ein WorkflowRun-Objekt.

  Die Informationen über die letzte Ausführung des Workflows.

- `Graph` – Ein WorkflowGraph-Objekt.

  Das Diagramm, das alle AWS Glue Komponenten, die zum Workflow gehören, als Knoten und gerichtete Verbindungen zwischen ihnen als Kanten darstellt.

- `CreationStatus` – UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `CREATED` | `CREATION_FAILED`).

  Der Erstellungsstatus des Workflows.

- `MaxConcurrentRuns` – Zahl (Ganzzahl).

  Mit diesem Parameter können Sie unerwünschte Mehrfachaktualisierungen von Daten verhindern, Kosten kontrollieren oder in einigen Fällen verhindern, dass die maximale Anzahl gleichzeitiger Durchläufe eines Komponentenauftrags überschritten wird. Wenn Sie für diesen Parameter keinen Wert angeben, ist die Anzahl der gleichzeitigen Workflow-Ausführungen nicht begrenzt.

- `BlueprintDetails` – Ein BlueprintDetails-Objekt.

  Diese Struktur gibt die Details des Blueprints an, aus dem dieser spezielle Workflow erstellt wird.

WorkflowGraph Struktur

Ein Workflow-Diagramm stellt den vollständigen Workflow mit allen im Workflow vorhandenen AWS Glue -Komponenten und alle gerichteten Verbindungen zwischen ihnen dar.

Felder

- `Nodes` – Ein Array mit Knoten-Objekten.

  Eine Liste der AWS Glue Komponenten, die zum Workflow gehören, wird als Knoten dargestellt.
- `Edges` – Ein Array mit Edge-Objekten.

  Eine Liste aller gerichteten Verbindungen zwischen den zum Workflow gehörenden Knoten.

WorkflowRun Struktur

Eine Workflow-Ausführung ist die Ausführung eines Workflow zur Bereitstellung aller Laufzeitinformationen.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Workflows, der ausgeführt wurde.
- `WorkflowRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID dieser Workflow-Ausführung.
- `PreviousRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der vorherigen Workflow-Ausführung.
- `WorkflowRunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die Workflow-Ausführungseigenschaften, die während der Ausführung festgelegt wurden.
- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Workflow-Ausführung gestartet wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Workflow vollständig ausgeführt wurde.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `COMPLETED` | `STOPPING` | `STOPPED` | `ERROR`).

  Der Status der Workflow-Ausführung.

- `ErrorMessage` – UTF-8-Zeichenfolge.

  Diese Fehlermeldung beschreibt alle Fehler, die beim Starten der Workflow-Ausführung aufgetreten sind. Derzeit ist die einzige Fehlermeldung „Concurrent runs exceeded for workflow: (Gleichzeitige Durchläufe für Workflow überschritten:) `foo`.“

- `Statistics` – Ein WorkflowRunStatistics-Objekt.

  Die Statistiken der Ausführung.

- `Graph` – Ein WorkflowGraph-Objekt.

  Das Diagramm, das alle AWS Glue Komponenten, die zum Arbeitsablauf gehören, als Knoten und gerichtete Verbindungen zwischen ihnen als Kanten darstellt.

- `StartingEventBatchCondition` – Ein StartingEventBatchCondition-Objekt.

  Die Batchbedingung, die den Workflow gestartet hat.

WorkflowRunStatistics Struktur

Workflow-Ausführungsstatistiken stellen Statistiken über die Workflow-Ausführung bereit.

Felder

- `TotalActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen in der Workflow-Ausführung.

- `TimeoutActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen, die das Zeitlimit überschritten haben.

- `FailedActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen, die fehlgeschlagen sind.

- `StoppedActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen, die gestoppt wurden.

- `SucceededActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen, die erfolgreich waren.

- `RunningActions` – Zahl (Ganzzahl).

  Gesamtzahl der Aktionen im Ausführungszustand.

- `ErroredActions` – Zahl (Ganzzahl).

  Gibt die Anzahl der Auftragsausführungen im ERROR-Status während der Ausführung des Workflows an.

- `WaitingActions` – Zahl (Ganzzahl).

  Gibt an, wie viele Aufträge in der Ausführung des Workflows im WAITING-Status ausgeführt werden.

StartingEventBatchCondition Struktur

Die Batchbedingung, die den Workflow gestartet hat. Entweder ist die Anzahl der Ereignisse in der Batchgröße eingetroffen, in diesem Fall ist das BatchSize Element ungleich Null, oder das Batchfenster ist abgelaufen, in diesem Fall ist das BatchWindow Element ungleich Null.

Felder

- `BatchSize` – Zahl (Ganzzahl).

  Anzahl der Ereignisse im Batch

- `BatchWindow` – Zahl (Ganzzahl).

  Dauer des Batchfensters in Sekunden.

Blueprint-Struktur

Die Details eines Blueprints.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.

- `Description` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Die Beschreibung des Blueprints.

- `CreatedOn` – Zeitstempel.

  Das Datum und die Uhrzeit der Registrierung des Blueprints.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit der letzten Änderung des Blueprints.

- `ParameterSpec` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 131 072 Bytes lang.

  Eine JSON-Zeichenfolge, welche die Liste der Parameterspezifikationen für den Blueprint angibt.

- `BlueprintLocation` – UTF-8-Zeichenfolge.

  Gibt den Pfad in Amazon S3 an, in dem der Blueprint veröffentlicht wird.

- `BlueprintServiceLocation` – UTF-8-Zeichenfolge.

  Gibt einen Pfad in Amazon S3 an, in den der Blueprint kopiert wird, wenn Sie `CreateBlueprint/UpdateBlueprint` aufrufen, um den Blueprint in AWS Glue zu registrieren.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `CREATING` | `ACTIVE` | `UPDATING` | `FAILED`).

  Der Status der Blueprint-Registrierung.

  - Creating (Wird erstellt) – Die Blueprint-Registrierung wird ausgeführt.
  - Active (Aktiv) – Der Blueprint wurde erfolgreich registriert.
  - Updating (wird aktualisiert) – Es wird eine Aktualisierung der Blueprint-Registrierung durchgeführt.
  - Failed (Fehlgeschlagen) – Die Blueprint-Registrierung ist fehlgeschlagen.

- `ErrorMessage` – UTF-8-Zeichenfolge.

  Eine Fehlermeldung.

- `LastActiveDefinition` – Ein LastActiveDefinition-Objekt.

Wenn mehrere Versionen eines Blueprints vorhanden sind und die neueste Version Fehler aufweist, gibt dieses Attribut die letzte erfolgreiche Blueprint-Definition an, die mit dem Dienst verfügbar ist.

BlueprintDetails Struktur

Die Details eines Blueprints.

Felder

- `BlueprintName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Ausführungs-ID für diesen Blueprint.

LastActiveDefinition Struktur

Wenn mehrere Versionen eines Blueprints vorhanden sind und die neueste Version Fehler aufweist, gibt dieses Attribut die letzte erfolgreiche Blueprint-Definition an, die mit dem Dienst verfügbar ist.

Felder

- `Description` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Die Beschreibung des Blueprints.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit der letzten Änderung des Blueprints.

- `ParameterSpec` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 131 072 Bytes lang.

  Eine JSON-Zeichenfolge, welche die Parameter für den Blueprint angibt.

- `BlueprintLocation` – UTF-8-Zeichenfolge.

Gibt einen Pfad in Amazon S3 an, in dem der Blueprint vom AWS Glue Entwickler veröffentlicht wird.

- `BlueprintServiceLocation` – UTF-8-Zeichenfolge.

  Gibt einen Pfad in Amazon S3 an, in den der Blueprint kopiert wird, wenn Sie den Blueprint erstellen oder aktualisieren.

BlueprintRun Struktur

Die Details einer Blueprint-Ausführung.

Felder

- `BlueprintName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Ausführungs-ID für diese Blueprint-Ausführung.

- `WorkflowName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name eines Workflows, der als Ergebnis einer erfolgreichen Blueprint-Ausführung erstellt wird. Weist eine Blueprint-Ausführung einen Fehler auf, wird kein Workflow erstellt.

- `State` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `SUCCEEDED` | `FAILED` | `ROLLING_BACK`).

  Der Status der Blueprint-Ausführung. Die möglichen Werte sind:

  - Running (Wird ausgeführt) – Der Blueprint wird ausgeführt.
  - Succeeded (Erfolgreich) – Die Blueprint-Ausführung wurde erfolgreich abgeschlossen.
  - Failed (Fehlgeschlagen) – Die Blueprint-Ausführung ist fehlgeschlagen und das Rollback ist abgeschlossen.
  - Rolling Back – Die Blueprint-Ausführung ist fehlgeschlagen und das Rollback wird ausgeführt.

- `StartedOn` – Zeitstempel.

Das Datum und die Uhrzeit, zu der die Blueprint-Ausführung gestartet wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Blueprint-Ausführung abgeschlossen wurde.

- `ErrorMessage` – UTF-8-Zeichenfolge.

  Gibt alle Fehler an, die beim Ausführen des Blueprints angezeigt werden.

- `RollbackErrorMessage` – UTF-8-Zeichenfolge.

  Wenn beim Erstellen der Entitäten eines Workflows Fehler auftreten, versuchen wir, die erstellten Entitäten bis zu diesem Punkt zurückzusetzen und zu löschen. Dieses Attribut gibt die Fehler an, die beim Versuch des Löschen der erstellten Entitäten auftreten.

- `Parameters` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 131 072 Bytes lang.

  Die Blueprint-Parameter als Zeichenfolge. Sie müssen für jeden Schlüssel einen Wert angeben, der aus der Parameterspezifikation erforderlich ist, die im `Blueprint$ParameterSpec` definiert ist.

- `RoleArn` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Custom string pattern #30.

  Der ARN der Rolle. Diese Rolle wird vom AWS Glue Service übernommen und zur Erstellung des Workflows und anderer Entitäten eines Workflows verwendet.

Operationen

- CreateWorkflow Aktion (Python: create_workflow)
- UpdateWorkflow Aktion (Python: update_workflow)
- DeleteWorkflow Aktion (Python: delete_workflow)
- GetWorkflow Aktion (Python: get_workflow)
- ListWorkflows Aktion (Python: list_workflows)
- BatchGetWorkflows Aktion (Python: batch_get_workflows)
- GetWorkflowRun Aktion (Python: get_workflow_run)
- GetWorkflowRuns Aktion (Python: get_workflow_runs)
- GetWorkflowRunProperties Aktion (Python: get_workflow_run_properties)

- PutWorkflowRunProperties Aktion (Python: put_workflow_run_properties)
- CreateBlueprint Aktion (Python: create_blueprint)
- UpdateBlueprint Aktion (Python: update_blueprint)
- DeleteBlueprint Aktion (Python: delete_blueprint)
- ListBlueprints Aktion (Python: list_blueprints)
- BatchGetBlueprints Aktion (Python: batch_get_blueprints)
- StartBlueprintRun Aktion (Python: start_blueprint_run)
- GetBlueprintRun Aktion (Python: get_blueprint_run)
- GetBlueprintRuns Aktion (Python: get_blueprint_runs)
- StartWorkflowRun Aktion (Python: start_workflow_run)
- StopWorkflowRun Aktion (Python: stop_workflow_run)
- ResumeWorkflowRun Aktion (Python: resume_workflow_run)

CreateWorkflow Aktion (Python: create_workflow)

Erstellt einen neuen Workflow.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name, der dem Workflow zugewiesen werden soll. Er sollte innerhalb Ihres Kontos eindeutig sein.
- `Description`— UTF-8-Zeichenfolge, nicht länger als 120000 Byte.

  Eine Beschreibung des Workflows.
- `DefaultRunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Sammlung von Eigenschaften, die als Teil einer jeden Ausführung des Workflows verwendet werden sollen.

Ausführungseigenschaften können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Eigenschaften. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie sie innerhalb der Workflow-Ausführung verwenden möchten.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die mit diesem Workflow zu verwendenden Tags.

- `MaxConcurrentRuns` – Zahl (Ganzzahl).

  Mit diesem Parameter können Sie unerwünschte Mehrfachaktualisierungen von Daten verhindern, Kosten kontrollieren oder in einigen Fällen verhindern, dass die maximale Anzahl gleichzeitiger Durchläufe eines Komponentenauftrags überschritten wird. Wenn Sie für diesen Parameter keinen Wert angeben, ist die Anzahl der gleichzeitigen Workflow-Ausführungen nicht begrenzt.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows, der als Teil der Anforderung bereitgestellt wurde.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateWorkflow Aktion (Python: update_workflow)

Aktualisiert einen vorhandenen Workflow.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des zu aktualisierenden Workflows.

- `Description`— UTF-8-Zeichenfolge, nicht länger als 120000 Byte.

  Die Beschreibung des Workflows.

- `DefaultRunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Eine Sammlung von Eigenschaften, die als Teil einer jeden Ausführung des Workflows verwendet werden sollen.

  Ausführungseigenschaften können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Eigenschaften. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie sie innerhalb der Workflow-Ausführung verwenden möchten.

- `MaxConcurrentRuns` – Zahl (Ganzzahl).

  Mit diesem Parameter können Sie unerwünschte Mehrfachaktualisierungen von Daten verhindern, Kosten kontrollieren oder in einigen Fällen verhindern, dass die maximale Anzahl gleichzeitiger Durchläufe eines Komponentenauftrags überschritten wird. Wenn Sie für diesen Parameter keinen Wert angeben, ist die Anzahl der gleichzeitigen Workflow-Ausführungen nicht begrenzt.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name des Workflows, der in der Eingabe angegeben wurde.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

DeleteWorkflow Aktion (Python: delete_workflow)

Löscht einen Workflow.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Workflows, der gelöscht werden soll.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows, der in der Eingabe angegeben wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

GetWorkflow Aktion (Python: get_workflow)

Ruft Ressourcenmetadaten für einen Workflow ab.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des abzurufenden Workflows.
- `IncludeGraph` – Boolesch.

  Gibt an, ob bei der Rückgabe der Workflow-Ressourcen-Metadaten ein Diagramm eingeschlossen werden soll.

Antwort

- `Workflow` – Ein Workflow-Objekt.

  Die Ressourcenmetadaten für den Workflow.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ListWorkflows Aktion (Python: list_workflows)

Listet die Namen der Workflows auf, die im Konto erstellt wurden.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 25.

Die maximale Größe der auszugebenden Liste.

Antwort

- `Workflows` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Liste der Namen von Workflows im Konto.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn nicht alle Workflow-Namen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetWorkflows Aktion (Python: batch_get_workflows)

Gibt eine Liste von Ressourcen-Metadaten für eine bestimmte Liste von Workflow-Namen zurück. Nach dem Aufrufen der `ListWorkflows`-Operation können Sie diese Operation aufrufen, um auf die Daten zuzugreifen, für die Ihnen Berechtigungen erteilt wurden. Dieser Vorgang unterstützt alle IAM-Berechtigungen, einschließlich Berechtigungsbedingungen, die Tags verwenden.

Anforderung

- `Names` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste von Workflow-Namen, die von der `ListWorkflows`-Operation als Namen zurückgegeben werden können.
- `IncludeGraph` – Boolesch.

  Gibt an, ob bei der Rückgabe der Workflow-Ressourcen-Metadaten ein Diagramm eingeschlossen werden soll.

Antwort

- `Workflows` – Ein Array mit Workflow-Objekten, nicht weniger als 1 und nicht mehr als 25 Strukturen.

  Eine Liste von Workflow-Ressourcen-Metadaten.
- `MissingWorkflows` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste der Namen von nicht gefundenen Workflows.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetWorkflowRun Aktion (Python: get_workflow_run)

Ruft die Metadaten für eine bestimmte Workflow-Ausführung ab. Der Verlauf der Auftragsausführung ist 90 Tage lang für Ihren Workflow und Ihre Auftragsausführung zugänglich.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des ausgeführten Workflows.
- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Workflow-Ausführung.
- `IncludeGraph` – Boolesch.

  Gibt an, ob das Workflow-Diagramm in die Antwort eingeschlossen werden soll.

Antwort

- `Run` – Ein WorkflowRun-Objekt.

  Die Metadaten der angeforderten Workflow-Ausführung.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRuns Aktion (Python: get_workflow_runs)

Ruft Metadaten für alle Ausführungen eines bestimmten Workflows ab.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Workflows, dessen Metadaten der Ausführungen zurückgegeben werden sollen.
- `IncludeGraph` – Boolesch.

  Gibt an, ob das Workflow-Diagramm in die Antwort eingeschlossen werden soll.
- `NextToken` – UTF-8-Zeichenfolge.

  Die maximale Größe der Antwort.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl von Workflow-Ausführungen, in die Antwort eingeschlossen werden sollen.

Antwort

- `Runs` – Ein Array mit WorkflowRun-Objekten, nicht weniger als 1 und nicht mehr als 1 000 Strukturen.

Eine Liste der Metadatenobjekte der Workflow-Ausführung.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, falls nicht alle angeforderten Workflow-Ausführungen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRunProperties Aktion (Python: get_workflow_run_properties)

Ruft die Workflow-Ausführungseigenschaften ab, die während der Ausführung festgelegt wurden.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Workflows, der ausgeführt wurde.

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Workflow-Ausführung, deren Ausführungseigenschaften zurückgegeben werden sollten.

Antwort

- `RunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

Die Workflow-Ausführungseigenschaften, die während der angegebenen Ausführung festgelegt wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

PutWorkflowRunProperties Aktion (Python: put_workflow_run_properties)

Legt die angegebenen Workflow-Ausführungseigenschaften für die angegebene Workflow-Ausführung fest. Wenn eine Eigenschaft für die angegebene Ausführung bereits vorhanden ist, wird der Wert überschrieben. Andernfalls wird die Eigenschaft zu den vorhandenen Eigenschaften hinzugefügt.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name des Workflows, der ausgeführt wurde.
- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Workflow-Ausführung, deren Ausführungseigenschaften aktualisiert werden sollten.
- `RunProperties` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die für die angegebene Ausführung festzulegenden Eigenschaften.

Ausführungseigenschaften können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Eigenschaften. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie sie innerhalb der Workflow-Ausführung verwenden möchten.

Antwort

- Keine Antwortparameter.

Fehler

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

CreateBlueprint Aktion (Python: create_blueprint)

Registriert einen Blueprint bei. AWS Glue

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.
- `Description` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Eine Beschreibung des Blueprints.
- `BlueprintLocation` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 8 192 Bytes lang, passend zum Custom string pattern #32.

  Gibt einen Pfad in Amazon S3 an, in dem der Blueprint veröffentlicht wird.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags, die auf diesen Blueprint angewendet werden sollen.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen des Blueprints zurück, der registriert wurde.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

UpdateBlueprint Aktion (Python: update_blueprint)

Aktualisiert einen registrierten Blueprint.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.
- `Description` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 512 Bytes lang.

  Eine Beschreibung des Blueprints.

- `BlueprintLocation` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 8 192 Bytes lang, passend zum Custom string pattern #32.

  Gibt einen Pfad in Amazon S3 an, in dem der Blueprint veröffentlicht wird.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen des Blueprints zurück, der aktualisiert wurde.

Fehler

- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `IllegalBlueprintStateException`

DeleteBlueprint Aktion (Python: delete_blueprint)

Löscht einen vorhandenen Blueprint.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Blueprints, der gelöscht werden soll.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Gibt den Namen des Blueprints zurück, der gelöscht wurde.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListBlueprints Aktion (Python: list_blueprints)

Listet alle Blueprint-Namen in einem Konto auf.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 25.

  Die maximale Größe der auszugebenden Liste.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Filtert die Liste nach einem AWS Ressourcen-Tag.

Antwort

- `Blueprints` – Ein UTF-8-Zeichenfolgen-Array.

  Liste der Namen von Blueprints im Konto.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn nicht alle Blueprint-Namen zurückgegeben wurden.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetBlueprints Aktion (Python: batch_get_blueprints)

Ruft Informationen zu einer Liste von Blueprints ab.

Anforderung

- `Names` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 25 Zeichenfolgen.

  Eine Liste von Blueprint-Namen.
- `IncludeBlueprint` – Boolesch.

  Gibt an, ob der Blueprint in die Antwort eingeschlossen werden soll.
- `IncludeParameterSpec` – Boolesch.

  Gibt an, ob die Parameter als JSON-Zeichenfolge für den Blueprint in die Antwort eingeschlossen werden sollen.

Antwort

- `Blueprints` – Ein Array mit Blueprint-Objekten.

  Gibt eine Blueprint-Liste als `Blueprints`-Objekt aus.
- `MissingBlueprints` – Ein UTF-8-Zeichenfolgen-Array.

  Gibt eine Liste von `BlueprintNames` aus, die nicht gefunden wurden.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartBlueprintRun Aktion (Python: start_blueprint_run)

Startet eine neue Ausführung des angegebenen Blueprints.

Anforderung

- `BlueprintName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.
- `Parameters` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 131 072 Bytes lang.

  Gibt die Parameter als `BlueprintParameters`-Objekt aus.
- `RoleArn` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Custom string pattern #30.

  Gibt die IAM-Rolle an, die zum Erstellen des Workflows verwendet wird.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Ausführungs-ID für diese Blueprint-Ausführung.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`
- `EntityNotFoundException`
- `IllegalBlueprintStateException`

GetBlueprintRun Aktion (Python: get_blueprint_run)

Ruft die Details einer Blueprint-Ausführung ab.

Anforderung

- `BlueprintName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #31.

  Der Name des Blueprints.

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Ausführungs-ID für den Blueprint, den Sie abrufen möchten.

Antwort

- `BlueprintRun` – Ein BlueprintRun-Objekt.

  Gibt ein `BlueprintRun`-Objekt zurück.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetBlueprintRuns Aktion (Python: get_blueprint_runs)

Ruft die Details von Blueprint-Ausführungen für den angegebenen Blueprint ab.

Anforderung

- `BlueprintName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Blueprints.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

Die maximale Größe der auszugebenden Liste.

Antwort

- `BlueprintRuns` – Ein Array mit BlueprintRun-Objekten.

  Gibt eine Liste von `BlueprintRun`-Objekten zurück.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn nicht alle Blueprint-Ausführungen zurückgegeben wurden.

Fehler

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartWorkflowRun Aktion (Python: start_workflow_run)

Startet eine neue Ausführung des angegebenen Workflows.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des zu startenden Workflows.

- `RunProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine UTF-8-Zeichenfolge.

  Die Workflow-Ausführungseigenschaften für die neue Workflow-Ausführung.

Ausführungseigenschaften können protokolliert werden. Übergeben Sie keine Klartext-Geheimnisse als Eigenschaften. Rufen Sie Geheimnisse aus einer AWS Glue Verbindung, einem AWS Secrets Manager oder einem anderen geheimen Verwaltungsmechanismus ab, wenn Sie sie innerhalb der Workflow-Ausführung verwenden möchten.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine ID für die neue Ausführung.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

StopWorkflowRun Aktion (Python: stop_workflow_run)

Beendet die Ausführung der angegebenen Workflow-Ausführung.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Workflows, der beendet werden soll.

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Workflow-Ausführung zum Beenden.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `IllegalWorkflowStateException`

ResumeWorkflowRun Aktion (Python: resume_workflow_run)

Startet ausgewählte Knoten einer vorherigen teilweise abgeschlossenen Workflow-Ausführung neu und setzt die Workflow-Ausführung fort. Dadurch werden die ausgewählten Knoten und alle den ausgewählten Knoten nachgelagerten Knoten ausgeführt.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des fortzusetzenden Workflows.

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der Workflow-Ausführung zum Fortsetzen.

- `NodeIds` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Eine Liste der Knoten IDs für die Knoten, die Sie neu starten möchten. Für die Knoten, die neu gestartet werden sollen, muss bei der ursprünglichen Ausführung ein Ausführungsversuch erfolgt sein.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die neue ID, die der wiederaufgenommenen Workflow-Ausführung zugewiesen wurde. Jede Fortsetzung einer Workflow-Ausführung hat eine neue Ausführungs-ID.
- `NodeIds` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der Knoten IDs für die Knoten, die tatsächlich neu gestartet wurden.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentRunsExceededException`
- `IllegalWorkflowStateException`

Nutzungsprofile

Die API für Nutzungsprofile beschreibt die Datentypen und die API im Zusammenhang mit der Erstellung, Aktualisierung oder Anzeige von Nutzungsprofilen in AWS Glue.

Datentypen

- ProfileConfiguration Struktur
- ConfigurationObject Struktur
- UsageProfileDefinition Struktur

ProfileConfiguration Struktur

Gibt die Job- und Sitzungswerte an, die ein Administrator in einem AWS Glue Nutzungsprofil konfiguriert.

Felder

- `SessionConfiguration` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist ein A ConfigurationObject-Objekt.

  Eine Schlüssel-Wert-Zuordnung von Konfigurationsparametern für Sitzungen. AWS Glue
- `JobConfiguration` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist ein A ConfigurationObject-Objekt.

  Eine Schlüssel-Wert-Zuordnung von Konfigurationsparametern für Jobs. AWS Glue

ConfigurationObject Struktur

Gibt die Werte an, die ein Administrator für jeden Job- oder Sitzungsparameter festlegt, der in einem AWS Glue Nutzungsprofil konfiguriert ist.

Felder

- `DefaultValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #35.

  Ein Standardwert für den Parameter.
- `AllowedValues` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste der zulässigen Werte für den Parameter.
- `MinValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #35.

  Ein zulässiger Mindestwert für den Parameter.
- `MaxValue` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang, passend zum Custom string pattern #35.

Ein zulässiger Höchstwert für den Parameter.

UsageProfileDefinition Struktur

Beschreibt ein AWS Glue Nutzungsprofil.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Nutzungsprofils.

- `CreatedOn` – Zeitstempel.

  Datum und Uhrzeit der Erstellung des Nutzungsprofils.

- `LastModifiedOn` – Zeitstempel.

  Datum und Uhrzeit der letzten Änderung des Nutzungsprofils.

Operationen

- CreateUsageProfile Aktion (Python: create_usage_profile)
- GetUsageProfile Aktion (Python: get_usage_profile)
- UpdateUsageProfile Aktion (Python: update_usage_profile)
- DeleteUsageProfile Aktion (Python: delete_usage_profile)
- ListUsageProfiles Aktion (Python: list_usage_profiles)

CreateUsageProfile Aktion (Python: create_usage_profile)

Erstellt ein Nutzungsprofil. AWS Glue

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Nutzungsprofils.

- `Configuration` – Erforderlich: Ein ProfileConfiguration-Objekt.

  Ein `ProfileConfiguration` Objekt, das die Job- und Sitzungswerte für das Profil angibt.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Eine Liste von Tags, die auf das Nutzungsprofil angewendet wurden.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils, das erstellt wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `OperationNotSupportedException`

GetUsageProfile Aktion (Python: get_usage_profile)

Ruft Informationen über das angegebene Nutzungsprofil ab. AWS Glue

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des abzurufenden Nutzungsprofils.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Nutzungsprofils.
- `Configuration` – Ein ProfileConfiguration-Objekt.

  Ein `ProfileConfiguration` Objekt, das die Job- und Sitzungswerte für das Profil angibt.
- `CreatedOn` – Zeitstempel.

  Datum und Uhrzeit der Erstellung des Nutzungsprofils.
- `LastModifiedOn` – Zeitstempel.

  Datum und Uhrzeit der letzten Änderung des Nutzungsprofils.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

UpdateUsageProfile Aktion (Python: update_usage_profile)

Aktualisieren Sie ein Nutzungsprofil. AWS Glue

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Nutzungsprofils.
- `Configuration` – Erforderlich: Ein ProfileConfiguration-Objekt.

  Ein `ProfileConfiguration` Objekt, das die Job- und Sitzungswerte für das Profil angibt.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Nutzungsprofils, das aktualisiert wurde.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`
- `ConcurrentModificationException`

DeleteUsageProfile Aktion (Python: delete_usage_profile)

Löscht das angegebene Nutzungsprofil. AWS Glue

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des zu löschenden Nutzungsprofils.

Antwort

- Keine Antwortparameter.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

ListUsageProfiles Aktion (Python: list_usage_profiles)

Listet alle Nutzungsprofile auf. AWS Glue

Anforderung

- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

  Ein Fortsetzungs-Token, der enthalten ist, wenn dies ein Fortsetzungsaufruf ist.
- `MaxResults`— Zahl (Ganzzahl), nicht weniger als 1 oder mehr als 200.

  Die maximale Anzahl von Nutzungsprofilen, die in einer einzigen Antwort zurückgegeben werden sollen.

Antwort

- `Profiles` – Ein Array mit UsageProfileDefinition-Objekten.

  Eine Liste von Nutzungsprofilobjekten (`UsageProfileDefinition`).
- `NextToken` – UTF-8-Zeichenfolge, nicht mehr als 400.000 Bytes lang.

Ein Fortsetzungs-Token, der vorhanden ist, wenn das aktuelle Listensegment nicht das letzte ist.

Fehler

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `OperationNotSupportedException`

Machine Learning API

Die Machine-Learning-API beschreibt die Machine-Learning-Datentypen und umfasst die API zum Erstellen, Löschen oder Aktualisieren einer Transformation oder zum Starten einer Machine-Learning-Aufgabe.

Datentypen

- TransformParameters Struktur
- EvaluationMetrics Struktur
- MLTransform Struktur
- FindMatchesParameters Struktur
- FindMatchesMetrics Struktur
- ConfusionMatrix Struktur
- GlueTable Struktur
- TaskRun Struktur
- TransformFilterCriteria Struktur
- TransformSortCriteria Struktur
- TaskRunFilterCriteria Struktur
- TaskRunSortCriteria Struktur
- TaskRunProperties Struktur
- FindMatchesTaskRunProperties Struktur

- ImportLabelsTaskRunProperties Struktur
- ExportLabelsTaskRunProperties Struktur
- LabelingSetGenerationTaskRunProperties Struktur
- SchemaColumn Struktur
- TransformEncryption Struktur
- MLUserDataEncryption Struktur
- ColumnImportance Struktur

TransformParameters Struktur

Die algorithmusspezifischen Parameter im Zusammenhang mit der Machine Learning-Transformation.

Felder

- `TransformType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `FIND_MATCHES`).

  Der Typ von Machine Learning-Transformation.

  Weitere Informationen zu den Typen von Machine Learning-Transformationen finden Sie unter Erstellen von Machine Learning-Transformationen.
- `FindMatchesParameters` – Ein FindMatchesParameters-Objekt.

  Die Parameter für den Algorithmus zur Suche nach Übereinstimmungen.

EvaluationMetrics Struktur

Auswertungsmetriken bieten eine Schätzung der Qualität Ihrer Machine Learning-Transformation.

Felder

- `TransformType` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `FIND_MATCHES`).

  Der Typ von Machine Learning-Transformation.
- `FindMatchesMetrics` – Ein FindMatchesMetrics-Objekt.

  Die Auswertungsmetriken für den Algorithmus zur Suche nach Übereinstimmungen.

MLTransform Struktur

Eine Struktur für eine Machine Learning-Transformation.

Felder

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Transform-ID, die für die Machine Learning-Transformation generiert wird. Die ID ist garantiert eindeutig und ändert sich nicht.
- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein benutzerdefinierter Name für die Machine Learning-Transformation. Namen sind nicht garantiert eindeutig und können jederzeit geändert werden.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine benutzerdefinierte Langform-Textbeschreibung für die Machine Learning-Transformation. Beschreibungen sind nicht garantiert eindeutig und können jederzeit geändert werden.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `NOT_READY` | `READY` | `DELETING`).

  Der aktuelle Status der Machine Learning-Transformation.
- `CreatedOn` – Zeitstempel.

  Ein Zeitstempel. Das Datum und die Uhrzeit, zu der diese Machine-Learning-Transformation erstellt wurde.
- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der Zeitpunkt, zu dem diese Machine-Learning-Transformation zuletzt geändert wurde.
- `InputRecordTables` – Ein Array mit GlueTable-Objekten, nicht mehr als 10 Strukturen.

  Eine Liste der von der Transformation verwendeten AWS Glue Tabellendefinitionen.
- `Parameters` – Ein TransformParameters-Objekt.

  Ein `TransformParameters`-Objekt. Sie können das Verhalten der Machine Learning-Transformation mithilfe von Parametern optimieren (anpassen), indem Sie angeben, von welchen

Daten sie lernt, sowie Ihre Vorzüge bezüglich verschiedener Verhältnisse (z. B. Präzision vs. Sensititivität oder Genauigkeit vs. Kosten).

- `EvaluationMetrics` – Ein EvaluationMetrics-Objekt.

  Ein `EvaluationMetrics`-Objekt. Auswertungsmetriken bieten eine Schätzung der Qualität Ihrer Machine Learning-Transformation.

- `LabelCount` – Zahl (Ganzzahl).

  Eine Zähl-ID für die von AWS Glue für diese Transformation generierten Bezeichnungsdateien. Während Sie eine bessere Transformation erstellen, können Sie die Labeling-Datei iterativ herunterladen, labeln und wieder hochladen.

- `Schema` – Ein Array mit SchemaColumn-Objekten, nicht mehr als 100 Strukturen.

  Eine Zuordnung von Schlüssel-Wert-Paaren, durch die die Spalten und Datentypen repräsentiert werden, für die diese Transformation ausgeführt werden kann. Hat eine Obergrenze von 100 Spalten.

- `Role` – UTF-8-Zeichenfolge.

  Der Name oder Amazon-Ressourcenname (ARN) der IAM-Rolle mit den erforderlichen Berechtigungen. Zu den erforderlichen Berechtigungen gehören sowohl AWS Glue Service-Rollenberechtigungen für AWS Glue Ressourcen als auch Amazon S3 S3-Berechtigungen, die für die Transformation erforderlich sind.

  - Diese Rolle benötigt AWS Glue Servicerollenberechtigungen, um den Zugriff auf Ressourcen in zu ermöglichen AWS Glue. Siehe Anfügen einer Richtlinie an IAM-Benutzer, die auf AWS Glue zugreifen.
  - Diese Rolle benötigt die Berechtigung für Ihre Amazon Simple Storage Service (Amazon S3)-Quellen, -Ziele, temporären Verzeichnisse und -Skripts sowie für beliebige Bibliotheken, die von der für diese Transformation ausgeführten Aufgabe genutzt werden.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Dieser Wert bestimmt, mit welcher Version AWS Glue dieser Transformation für maschinelles Lernen kompatibel ist. Glue 1.0 wird für die meisten Kunden empfohlen. Wenn der Wert nicht festgelegt ist, wird die Glue-Kompatibilität standardmäßig auf Glue 0.9 gesetzt. Weitere Informationen finden Sie unter AWS Glue -Versionen im Entwicklerhandbuch.

- `MaxCapacity` – Nummer (doppelt).

Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die Aufgabenausführungen für diese Transformation zugewiesen sind. Sie können 2 bis 100 zuweisen. DPUs Die Standardeinstellung ist 10. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

`MaxCapacity` ist eine sich mit `NumberOfWorkers` und `WorkerType` ausschließende Option.

- Wenn `NumberOfWorkers` oder `WorkerType` festgelegt ist, kann `MaxCapacity` nicht festgelegt werden.
- Wenn `MaxCapacity` festgelegt ist, kann weder `NumberOfWorkers` noch `WorkerType` festgelegt werden.
- Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).
- Für `MaxCapacity` und `NumberOfWorkers` muss der Wert mindestens 1 lauten.

Wenn das Feld `WorkerType` auf einen anderen Wert als `Standard` gesetzt ist, wird das Feld `MaxCapacity` automatisch eingestellt und wird schreibgeschützt.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der zugewiesen wird, wenn eine Aufgabe dieser Transformation ausgeführt wird. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 64 GB Festplattenplatz sowie 1 Executor pro Worker bereit.
  - Für den Worker-Typ `G.2X` stellt jeder Worker 8 vCPUs, 32 GB Speicher und 128 GB Festplattenplatz sowie 1 Executor pro Worker bereit.

  `MaxCapacity` ist eine sich mit `NumberOfWorkers` und `WorkerType` ausschließende Option.

  - Wenn `NumberOfWorkers` oder `WorkerType` festgelegt ist, kann `MaxCapacity` nicht festgelegt werden.
  - Wenn `MaxCapacity` festgelegt ist, kann weder `NumberOfWorkers` noch `WorkerType` festgelegt werden.
  - Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).
  - Für `MaxCapacity` und `NumberOfWorkers` muss der Wert mindestens 1 lauten.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl von Workern eines definierten `workerType`, die zugewiesen werden, wenn eine Aufgabe der Transformation ausgeführt wird.

  Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Zeitlimit der Machine Learning-Transformation in Minuten.

- `MaxRetries` – Zahl (Ganzzahl).

  Die maximale Anzahl der Wiederholungsversuche, nachdem eine `MLTaskRun` der Machine Learning-Transformation fehlgeschlagen ist.

- `TransformEncryption` – Ein TransformEncryption-Objekt.

  Die encryption-at-rest Einstellungen der Transformation, die für den Zugriff auf Benutzerdaten gelten. Machine-Learning-Transformationen können mithilfe von KMS auf in Amazon S3 verschlüsselte Benutzerdaten zugreifen.

FindMatchesParameters Struktur

Die Parameter zum Konfigurieren der Transformation zur Suche nach Übereinstimmungen.

Felder

- `PrimaryKeyColumnName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Single-line string pattern.

  Der Name einer Spalte, in der die Zeilen in der Quelltabelle eindeutig identifiziert werden. Dient zur leichteren Identifizierung übereinstimmender Datensätze.

- `PrecisionRecallTradeoff` – Zahl (Double), nicht mehr als 1,0.

  Der ausgewählte Wert, wenn Sie Ihre Transformation für ein ausgewogenes Verhältnis zwischen Präzision und Sensitivität optimieren. Ein Wert von 0,5 bedeutet keine Präferenz, ein Wert von 1.0 bedeutet ein Bias ausschließlich für Präzision und ein Wert von 0,0 bedeutet ein Bias für Sensitivität. Da dies ein Verhältnis ist, bedeutet die Wahl von Werten nahe 1,0 eine sehr geringe Sensitivität und die Wahl von Werten nahe 0,0 eine sehr geringe Präzision.

Die Präzisionsmetrik zeigt an, wie oft Ihr Modell korrekt ist, wenn es eine Übereinstimmung voraussagt.

Die Sensitivitätsmetrik gibt für eine tatsächliche Übereinstimmung an, wie oft das Modell die Übereinstimmung vorhersagt.

- `AccuracyCostTradeoff` – Zahl (Double), nicht mehr als 1,0.

  Der Wert, der ausgewählt wird, wenn Sie Ihre Transformation für ein ausgewogenes Verhältnis zwischen Genauigkeit und Kosten optimieren. Ein Wert von 0,5 bedeutet ein ausgewogenes Verhältnis zwischen Genauigkeit und Kosten. Ein Wert von 1.0 bedeutet ein Bias ausschließlich für Genauigkeit. Dies ist in der Regel mit höheren Kosten verbunden, manchmal mit wesentlich höheren Kosten. Ein Wert von 0,0 bedeutet ein Bias ausschließlich für Kosten. Dies führt zu einer weniger genauen `FindMatches`-Transformation, manchmal mit unannehmbarer Genauigkeit.

  Mit der Genauigkeit wird gemessen, wie gut die Transformation bei der Suche nach wahren positiven und wahren negativen Ergebnisse abschneidet. Eine Erhöhung der Genauigkeit erfordert mehr maschinelle Ressourcen und erhöht die Kosten. Sie resultiert aber auch in einer erhöhten Sensitivität.

  Mit den Kosten wird gemessen, wie viele Datenverarbeitungsressourcen und damit Gelder für die Ausführung der Transformation verbraucht werden.

- `EnforceProvidedLabels` – Boolesch.

  Der zu aktivierende oder zu deaktivierende Wert, um die Ausgabe entsprechend den von Benutzern bereitgestellten Labels zu erzwingen. Bei dem Wert `True` erzwingt die Transformation `find matches` die Ausgabe entsprechend den bereitgestellten Labels. Die Ergebnisse überschreiben die normalen Zusammenführungsergebnisse. Bei dem Wert `False` stellt die Transformation `find matches` nicht sicher, dass alle bereitgestellten Labels berücksichtigt werden. Die Ergebnisse sind vom trainierten Modell abhängig.

  Beachten Sie, dass sich die Konflations-Ausführungszeit erhöhen kann, wenn als Wert diese Einstellung „true“ festgelegt wird.

FindMatchesMetrics Struktur

Die Auswertungsmetriken für den Algorithmus zur Suche nach Übereinstimmungen. Die Qualität Ihrer Machine Learning-Transformation wird gemessen, indem Sie Ihre Transformation dazu

veranlassen, einige Übereinstimmungen vorauszusagen, und die Ergebnisse dann mit bekannten Übereinstimmungen aus demselben Datensatz vergleichen. Die Qualitätsmetriken basieren auf einer Untermenge Ihrer Daten. Sie sind also nicht präzise.

Felder

- `AreaUnderPRCurve` – Zahl (Double), nicht mehr als 1,0.

  Die Fläche unter der precision/recall Kurve (AUPRC) ist eine einzelne Zahl, die die Gesamtqualität der Transformation angibt. Diese Zahl ist unabhängig von der Wahl zwischen Präzision und Erinnerungsvermögen. Höhere Werte weisen auf ein attraktiveres Verhältnis zwischen Präzision und Sensitivität hin.

  Weitere Informationen finden Sie unter Precision and recall in Wikipedia.
- `Precision` – Zahl (Double), nicht mehr als 1,0.

  Die Präzisionsmetrik zeigt an, wie oft Ihre Transformation korrekt ist, wenn sie eine Übereinstimmung voraussagt. Insbesondere misst sie, wie gut die Transformation wahre positive Ergebnisse unter den insgesamt möglichen wahren positiven Ergebnissen ermittelt.

  Weitere Informationen finden Sie unter Precision and recall in Wikipedia.
- `Recall` – Zahl (Double), nicht mehr als 1,0.

  Die Sensitivitätsmetrik gibt für eine tatsächliche Übereinstimmung an, wie oft die Transformation die Übereinstimmung vorhersagt. Insbesondere misst sie, wie gut die Transformation die tatsächlichen positiven Ergebnisse aus der Gesamtzahl der Datensätze in den Quelldaten ermittelt.

  Weitere Informationen finden Sie unter Precision and recall in Wikipedia.
- `F1` – Zahl (Double), nicht mehr als 1,0.

  Die Metrik des maximalen F1-Wertes gibt die Genauigkeit der Transformation zwischen 0 und 1 an, wobei 1 ist die beste Genauigkeit ist.

  Weitere Informationen finden Sie unter F1 score in Wikipedia.
- `ConfusionMatrix` – Ein ConfusionMatrix-Objekt.

  Die Konfusionsmatrix zeigt Ihnen, was von Ihrer Transformation richtig vorausgesagt wird und welche Arten von Fehlern gemacht werden.

  Weitere Informationen finden Sie unter Confusion Matrix in Wikipedia.

- `ColumnImportances` – Ein Array mit ColumnImportance-Objekten, nicht mehr als 100 Strukturen.

  Eine Liste von `ColumnImportance`-Strukturen mit Metriken für die Spaltenbedeutung, sortiert nach absteigender Wichtigkeit.

ConfusionMatrix Struktur

Die Konfusionsmatrix zeigt Ihnen, was von Ihrer Transformation richtig vorausgesagt wird und welche Arten von Fehlern gemacht werden.

Weitere Informationen finden Sie unter Confusion Matrix in Wikipedia.

Felder

- `NumTruePositives` – Zahl (lang).

  Die Anzahl der Übereinstimmungen in den Daten, die von der Transformation korrekt gefunden wurden, in der Konfusionsmatrix für Ihre Transformation.
- `NumFalsePositives` – Zahl (lang).

  Die Anzahl von Nichtübereinstimmungen in den Daten, die von der Transformation fälschlicherweise als Übereinstimmung klassifiziert wurden, in der Konfusionsmatrix für Ihre Transformation.
- `NumTrueNegatives` – Zahl (lang).

  Die Anzahl von Nichtübereinstimmungen in den Daten, die von der Transformation korrekt abgelehnt wurden, in der Konfusionsmatrix für Ihre Transformation.
- `NumFalseNegatives` – Zahl (lang).

  Die Anzahl der Übereinstimmungen in den Daten, die von der Transformation nicht gefunden wurden, in der Konfusionsmatrix für Ihre Transformation.

GlueTable Struktur

Die Datenbank und Tabelle in der AWS Glue Data Catalog , die für Eingabe- oder Ausgabedaten verwendet werden.

Felder

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Datenbankname im AWS Glue Data Catalog.

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Tabellenname im AWS Glue Data Catalog.

- `CatalogId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für AWS Glue Data Catalog.

- `ConnectionName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der zu löschenden Verbindung zu AWS Glue Data Catalog.

- `AdditionalOptions` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht weniger als 1 oder mehr als 10 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine Beschreibungszeichenfolge, die nicht länger als 2 048 Byte ist und mit URI address multi-line string pattern übereinstimmt.

  Zusätzliche Optionen für die Tabelle. Derzeit werden zwei Schlüssel unterstützt:

  - `pushDownPredicate`: zum Filtern nach Partitionen, ohne alle Dateien in Ihrem Datensatz auflisten und lesen zu müssen.
  - `catalogPartitionPredicate`: zur Verwendung des serverseitigen Partition-Pruning mithilfe von Partitionsindizes in der AWS Glue Data Catalog.

TaskRun Struktur

Die Sampling-Parameter im Zusammenhang mit der Machine Learning-Transformation.

Felder

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung für die Transformation.

- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung für diese Aufgabenausführung.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der aktuelle Status der angeforderten Aufgabenausführung.

- `LogGroupName` – UTF-8-Zeichenfolge.

  Die Namen der Protokollgruppe für die sichere Protokollierung, dieser Aufgabenausführung zugeordnet ist.

- `Properties` – Ein TaskRunProperties-Objekt.

  Gibt die dieser Aufgabenausführung zugeordneten Konfigurationseigenschaften an.

- `ErrorString` – UTF-8-Zeichenfolge.

  Die Liste der Fehlerzeichenfolgen im Zusammenhang mit dieser Aufgabenausführung.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Aufgabenausführung gestartet wurde.

- `LastModifiedOn` – Zeitstempel.

  Der Zeitpunkt, an dem die angeforderte Aufgabenausführung zuletzt aktualisiert wurde.

- `CompletedOn` – Zeitstempel.

  Der Zeitpunkt, an dem die angeforderte Aufgabenausführung zuletzt abgeschlossen wurde.

- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Aufgabenausführung Ressourcen verbraucht wurden.

TransformFilterCriteria Struktur

Die Kriterien zum Filtern der Machine Learning-Transformationen.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutiger Transformationsname, der zum Filtern der Machine Learning-Transformationen verwendet wird.
- `TransformType` – UTF-8-Zeichenfolge (zulässige Werte: `FIND_MATCHES`).

  Der Typ der Machine Learning-Transformation, der zum Filtern der Machine-Learning-Transformationen verwendet wird.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `NOT_READY` | `READY` | `DELETING`).

  Filtert die Liste der Machine Learning-Transformationen nach dem letzten bekannten Status der Transformationen (um anzugeben, ob eine Transformation verwendet werden kann). Möglich sind „NOT_READY“, „READY“ oder „DELETING“.
- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Dieser Wert bestimmt, mit welcher Version AWS Glue dieser Transformation für maschinelles Lernen kompatibel ist. Glue 1.0 wird für die meisten Kunden empfohlen. Wenn der Wert nicht festgelegt ist, wird die Glue-Kompatibilität standardmäßig auf Glue 0.9 gesetzt. Weitere Informationen finden Sie unter AWS Glue -Versionen im Entwicklerhandbuch.
- `CreatedBefore` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Transformationen erstellt wurden.
- `CreatedAfter` – Zeitstempel.

  Das Datum und die Uhrzeit, nach der die Transformationen erstellt wurden.
- `LastModifiedBefore` – Zeitstempel.

  Filtert nach Transformationen, die zuletzt vor diesem Datum geändert wurden.
- `LastModifiedAfter` – Zeitstempel.

  Filter nach Transformationen, die zuletzt nach diesem Datum gefiltert wurden.

- `Schema` – Ein Array mit SchemaColumn-Objekten, nicht mehr als 100 Strukturen.

  Filtert nach Datensätzen mit einem bestimmten Schema. Das Objekt `Map<Column, Type>`ist ein Array von Schlüssel-Wert-Paaren, die das von dieser Transformation akzeptierte Schema repräsentieren. Dabei ist `Column` der Name einer Spalte und `Type` der Datentyp, z. B. eine Ganzzahl oder eine Zeichenfolge. Hat eine Obergrenze von 100 Spalten.

TransformSortCriteria Struktur

Die Sortierkriterien im Zusammenhang mit der Machine Learning-Transformation.

Felder

- `Column` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `NAME` | `TRANSFORM_TYPE` | `STATUS` | `CREATED` | `LAST_MODIFIED`).

  Die in den Sortierkriterien im Zusammenhang mit der Machine Learning-Transformation zu verwendende Spalte.
- `SortDirection` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `DESCENDING` | `ASCENDING`).

  Die in den Sortierkriterien im Zusammenhang mit der Machine Learning-Transformation zu verwendende Sortierreihenfolge.

TaskRunFilterCriteria Struktur

Die Kriterien, die zum Filtern der Aufgabenausführungen für die Machine Learning-Transformation verwendet werden.

Felder

- `TaskRunType` – UTF-8-Zeichenfolge (zulässige Werte: `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Der Typ der Aufgabenausführung.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der aktuelle Status der Aufgabenausführung.

- `StartedBefore` – Zeitstempel.

  Filtert nach Aufgabenausführungen, die vor diesem Datum gestartet wurden.
- `StartedAfter` – Zeitstempel.

  Filtert nach Aufgabenausführungen, die nach diesem Datum gestartet wurden.

TaskRunSortCriteria Struktur

Die Sortierkriterien, die zum Sortieren der Liste von Aufgabenausführungen für die Machine Learning-Transformation verwendet werden.

Felder

- `Column` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `TASK_RUN_TYPE` | `STATUS` | `STARTED`).

  Die Spalte, die zum Sortieren der Liste von Aufgabenausführungen für die Machine Learning-Transformation verwendet werden soll.
- `SortDirection` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `DESCENDING` | `ASCENDING`).

  Die Sortierreihenfolge, die zum Sortieren der Liste von Aufgabenausführungen für die Machine Learning-Transformation verwendet werden soll.

TaskRunProperties Struktur

Die Konfigurationseigenschaften für die Aufgabenausführung.

Felder

- `TaskType` – UTF-8-Zeichenfolge (zulässige Werte: `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Der Typ der Aufgabenausführung.
- `ImportLabelsTaskRunProperties` – Ein ImportLabelsTaskRunProperties-Objekt.

  Die Konfigurationseigenschaften für eine Aufgabenausführung zum Importieren von Labels.
- `ExportLabelsTaskRunProperties` – Ein ExportLabelsTaskRunProperties-Objekt.

Die Konfigurationseigenschaften für eine Aufgabenausführung zum Exportieren von Labels.

- `LabelingSetGenerationTaskRunProperties` – Ein LabelingSetGenerationTaskRunProperties-Objekt.

  Die Konfigurationseigenschaften für eine Aufgabenausführung zum Generieren eines Labeling-Satzes.

- `FindMatchesTaskRunProperties` – Ein FindMatchesTaskRunProperties-Objekt.

  Die Konfigurationseigenschaften für eine Aufgabenausführung zur Suche nach Übereinstimmungen.

FindMatchesTaskRunProperties Struktur

Gibt Konfigurationseigenschaften für eine Aufgabenausführung zur Suche nach Übereinstimmungen an.

Felder

- `JobId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftrags-ID für die Aufgabenausführung zur Suche nach Übereinstimmungen.

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name, der dem Auftrag für die Aufgabenausführung zur Suche nach Übereinstimmungen zugewiesen ist.

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftragsausführungs-ID für die Aufgabenausführung zur Suche nach Übereinstimmungen.

ImportLabelsTaskRunProperties Struktur

Gibt die Konfigurationseigenschaften für eine Aufgabenausführung zum Importieren von Labels an.

Felder

- `InputS3Path` – UTF-8-Zeichenfolge.

  Der Amazon Simple Storage Service (Amazon S3)-Pfad, aus dem Sie Labels importieren.
- `Replace` – Boolesch.

  Gibt an, ob Ihre vorhandenen Bezeichnungen überschrieben werden sollen.

ExportLabelsTaskRunProperties Struktur

Gibt die Konfigurationseigenschaften für eine Aufgabenausführung zum Exportieren von Bezeichnungen an.

Felder

- `OutputS3Path` – UTF-8-Zeichenfolge.

  Der Amazon Simple Storage Service (Amazon S3) -Pfad, in den Sie die Bezeichnungen exportieren.

LabelingSetGenerationTaskRunProperties Struktur

Gibt Konfigurationseigenschaften für eine Aufgabenausführung zum Generieren eines Labeling-Satzes an.

Felder

- `OutputS3Path` – UTF-8-Zeichenfolge.

  Der Amazon Simple Storage Service (Amazon S3)-Pfad, in dem Sie den Labeling-Satz generieren.

SchemaColumn Struktur

Ein Schlüssel-Wert-Paar, durch das eine Spalte und ein Datentyp repräsentiert werden, für die diese Transformation ausgeführt werden kann. Der Parameter `Schema` der `MLTransform` kann bis zu 100 dieser Strukturen enthalten.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum Single-line string pattern.

  Der Name der Spalte.
- `DataType` – UTF-8-Zeichenfolge, nicht mehr als 131 072 Bytes lang, passend zum Single-line string pattern.

  Die Art von Daten in der Spalte.

TransformEncryption Struktur

Die encryption-at-rest Einstellungen der Transformation, die für den Zugriff auf Benutzerdaten gelten. Machine-Learning-Transformationen können mithilfe von KMS auf in Amazon S3 verschlüsselte Benutzerdaten zugreifen.

Darüber hinaus können importierte Labels und trainierte Transformationen jetzt mit einem vom Kunden bereitgestellten KMS-Schlüssel verschlüsselt werden.

Felder

- `MlUserDataEncryption` – Ein MLUserDataEncryption-Objekt.

  Ein `MLUserDataEncryption`-Objekt, das den Verschlüsselungsmodus und die vom Kunden bereitgestellte KMS-Schlüssel-ID enthält.
- `TaskRunSecurityConfigurationName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Gibt den Namen der Sicherheitskonfiguration an.

MLUserDataEncryption Struktur

Die encryption-at-rest Einstellungen der Transformation, die für den Zugriff auf Benutzerdaten gelten.

Felder

- `MlUserDataEncryptionMode` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `DISABLED` | `SSE-KMS="SSEKMS"`).

Der Verschlüsselungsmodus, der auf Benutzerdaten angewendet wird. Gültige Werte für sind:

- DEAKTIVIERT: Verschlüsselung ist deaktiviert
- SSEKMS: Verwendung der serverseitigen Verschlüsselung mit AWS Key Management Service (SSE-KMS) für in Amazon S3 gespeicherte Benutzerdaten.

- `KmsKeyId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID für den vom Kunden bereitgestellten KMS-Schlüssel.

ColumnImportance Struktur

Eine Struktur, die den Spaltennamen und die Spaltenbedeutung für eine Spalte enthält.

Anhand der Spaltenbedeutung können Sie verstehen, wie Spalten zu Ihrem Modell beitragen, indem Sie ermitteln, welche Spalten in Ihren Datensätzen wichtiger als andere sind.

Felder

- `ColumnName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name einer Spalte.
- `Importance` – Zahl (Double), nicht mehr als 1,0.

  Die Bewertung der Spaltenbedeutung als Dezimalzahl.

Operationen

- MLTransform Aktion erstellen (Python: create_ml_transform)
- MLTransform Aktion aktualisieren (Python: update_ml_transform)
- MLTransform Aktion löschen (Python: delete_ml_transform)
- MLTransform Aktion abrufen (Python: get_ml_transform)
- MLTransforms Aktion abrufen (Python: get_ml_transforms)
- MLTransforms Aktion auflisten (Python: list_ml_transforms)
- MLEvaluationTaskRun Aktion starten (Python: start_ml_evaluation_task_run)

- MLLabelingSetGenerationTaskRun Aktion starten (Python: start_ml_labeling_set_generation_task_run)
- Aktion „MLTaskAusführen abrufen“ (Python: get_ml_task_run)
- Aktion „Get MLTask Runs“ (Python: get_ml_task_runs)
- Aktion „MLTaskAusführen“ abbrechen (Python: cancel_ml_task_run)
- StartExportLabelsTaskRun Aktion (Python: start_export_labels_task_run)
- StartImportLabelsTaskRun Aktion (Python: start_import_labels_task_run)

MLTransform Aktion erstellen (Python: create_ml_transform)

Erzeugt eine Transformation für maschinelles Lernen AWS Glue . Diese Operation erstellt die Transformation und alle erforderlichen Parameter, um sie zu trainieren.

Rufen Sie diese Operation als ersten Schritt bei dem Prozess für die Deduplizierung von Daten mithilfe einer Machine Learning-Transformation (z. B. der `FindMatches`-Transformation) auf. Zusätzlich zu den Parametern, die Sie für Ihren Algoritmus verwenden möchten, können Sie eine optionale `Description` angeben.

Sie müssen auch bestimmte Parameter für die Aufgaben angeben, die in Ihrem Namen AWS Glue ausgeführt werden, um aus Ihren Daten zu lernen und eine hochwertige Transformation für maschinelles Lernen zu erstellen. Zu diesen Parametern gehören `Role` und optional `AllocatedCapacity`, `Timeout` und `MaxRetries`. .Weitere Informationen hierzu finden Sie unter Aufträge

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der eindeutige Name, den Sie der Transformation beim Erstellen geben.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der zu definierenden Machine Learning-Transformation. Der Standardwert ist eine leere Zeichenfolge.
- `InputRecordTables` – Erforderlich: Ein Array mit GlueTable-Objekten, nicht mehr als 10 Strukturen.

Eine Liste der von der Transformation verwendeten AWS Glue Tabellendefinitionen.

- `Parameters` – Erforderlich: Ein TransformParameters-Objekt.

  Die algorithmischen Parameter, die für den verwendeten Transformationstyp spezifisch sind. Bedingt abhängig vom Transformationstyp.

- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Der Name oder Amazon-Ressourcenname (ARN) der IAM-Rolle mit den erforderlichen Berechtigungen. Zu den erforderlichen Berechtigungen gehören sowohl AWS Glue Service-Rollenberechtigungen für AWS Glue Ressourcen als auch Amazon S3 S3-Berechtigungen, die für die Transformation erforderlich sind.

  - Diese Rolle benötigt AWS Glue Servicerollenberechtigungen, um den Zugriff auf Ressourcen in zu ermöglichen AWS Glue. Siehe Anfügen einer Richtlinie an IAM-Benutzer, die auf AWS Glue zugreifen.
  - Diese Rolle benötigt die Berechtigung für Ihre Amazon Simple Storage Service (Amazon S3)-Quellen, -Ziele, temporären Verzeichnisse und -Skripts sowie für beliebige Bibliotheken, die von der für diese Transformation ausgeführten Aufgabe genutzt werden.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Dieser Wert bestimmt, mit welcher Version AWS Glue dieser Transformation für maschinelles Lernen kompatibel ist. Glue 1.0 wird für die meisten Kunden empfohlen. Wenn der Wert nicht festgelegt ist, wird die Glue-Kompatibilität standardmäßig auf Glue 0.9 gesetzt. Weitere Informationen finden Sie unter AWS Glue -Versionen im Entwicklerhandbuch.

- `MaxCapacity` – Nummer (doppelt).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die Aufgabenausführungen für diese Transformation zugewiesen sind. Sie können 2 bis 100 zuweisen. DPUs Die Standardeinstellung ist 10. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  `MaxCapacity` ist eine sich mit `NumberOfWorkers` und `WorkerType` ausschließende Option.

  - Wenn `NumberOfWorkers` oder `WorkerType` festgelegt ist, kann `MaxCapacity` nicht festgelegt werden.

- Wenn `MaxCapacity` festgelegt ist, kann weder `NumberOfWorkers` noch `WorkerType` festgelegt werden.
- Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).
- Für `MaxCapacity` und `NumberOfWorkers` muss der Wert mindestens 1 lauten.

Wenn das Feld `WorkerType` auf einen anderen Wert als `Standard` gesetzt ist, wird das Feld `MaxCapacity` automatisch eingestellt und wird schreibgeschützt.

Wenn das Feld `WorkerType` auf einen anderen Wert als `Standard` gesetzt ist, wird das Feld `MaxCapacity` automatisch eingestellt und wird schreibgeschützt.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der zugeordnet wird, wenn diese Aufgabe ausgeführt wird. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 64 GB Festplattenplatz sowie 1 Executor pro Worker bereit.
  - Für den Worker-Typ `G.2X` stellt jeder Worker 8 vCPUs, 32 GB Speicher und 128 GB Festplattenplatz sowie 1 Executor pro Worker bereit.

  `MaxCapacity` ist eine sich mit `NumberOfWorkers` und `WorkerType` ausschließende Option.

  - Wenn `NumberOfWorkers` oder `WorkerType` festgelegt ist, kann `MaxCapacity` nicht festgelegt werden.
  - Wenn `MaxCapacity` festgelegt ist, kann weder `NumberOfWorkers` noch `WorkerType` festgelegt werden.
  - Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).
  - Für `MaxCapacity` und `NumberOfWorkers` muss der Wert mindestens 1 lauten.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType`, die zugewiesen werden, wenn diese Aufgabe ausgeführt wird.

  Wenn `WorkerType` festgelegt ist, ist `NumberOfWorkers` erforderlich (und umgekehrt).
- `Timeout` – Zahl (ganze Zahl), mindestens 1.

Das Timeout der Aufgabenausführung für diese Transformation in Minuten. Dies ist die maximale Zeit, für die eine Aufgabenausführung dieser Transformation Ressourcen verbrauchen kann, bevor sie beendet wird und in den Status `TIMEOUT` übergeht. Der Standardwert beträgt 2 880 Minuten (48 Stunden).

- `MaxRetries` – Zahl (Ganzzahl).

  Die maximale Anzahl an Versuchen, eine Aufgabe für diese Transformation erneut zu versuchen, nachdem eine Aufgabenausführung fehlschlägt.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Die Tags, die mit dieser Machine Learning-Transformation verwendet werden sollen. Sie können Tags verwenden, um den Zugriff auf die Machine Learning-Transformation zu beschränken. Weitere Informationen zu Tags in AWS Glue finden Sie unter AWS Tags in AWS Glue im Entwicklerhandbuch.

- `TransformEncryption` – Ein TransformEncryption-Objekt.

  Die encryption-at-rest Einstellungen der Transformation, die für den Zugriff auf Benutzerdaten gelten. Machine-Learning-Transformationen können mithilfe von KMS auf in Amazon S3 verschlüsselte Benutzerdaten zugreifen.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung, die für die Transformation generiert wird.

Fehler

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`

- `InternalServiceException`
- `AccessDeniedException`
- `ResourceNumberLimitExceededException`
- `IdempotentParameterMismatchException`

MLTransform Aktion aktualisieren (Python: update_ml_transform)

Aktualisiert eine bestehende Machine Learning-Transformation. Rufen Sie diese Operation zur Optimierung der Algorithmusparameter auf, um bessere Ergebnisse zu erzielen.

Nach dem Aufruf dieser Operation können Sie die Operation `StartMLEvaluationTaskRun` aufrufen, um zu bewerten, wie gut Ihre Ziele (z. B. Verbessern der Qualität oder Kostenwirksamkeit Ihrer Machine Learning-Transformation) mit Ihren neuen Parametern erreicht wurden.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung, die beim Erstellen der Transformation generiert wurde.
- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der eindeutige Name, den Sie der Transformation beim Erstellen gegeben haben.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Transformation. Der Standardwert ist eine leere Zeichenfolge.
- `Parameters` – Ein TransformParameters-Objekt.

  Die Konfigurationsparameter, die für den verwendeten Transformationstyp (Algorithmus) spezifisch sind. Bedingt abhängig vom Transformationstyp.
- `Role` – UTF-8-Zeichenfolge.

  Der Name oder Amazon-Ressourcenname (ARN) der IAM-Rolle mit den erforderlichen Berechtigungen.
- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

Dieser Wert bestimmt, mit welcher Version AWS Glue dieser Transformation für maschinelles Lernen kompatibel ist. Glue 1.0 wird für die meisten Kunden empfohlen. Wenn der Wert nicht festgelegt ist, wird die Glue-Kompatibilität standardmäßig auf Glue 0.9 gesetzt. Weitere Informationen finden Sie unter AWS Glue -Versionen im Entwicklerhandbuch.

- `MaxCapacity` – Nummer (doppelt).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die Aufgabenausführungen für diese Transformation zugewiesen sind. Sie können 2 bis 100 zuweisen. DPUs Die Standardeinstellung ist 10. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Wenn das Feld `WorkerType` auf einen anderen Wert als `Standard` gesetzt ist, wird das Feld `MaxCapacity` automatisch eingestellt und wird schreibgeschützt.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der zugeordnet wird, wenn diese Aufgabe ausgeführt wird. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 64 GB Festplattenplatz sowie 1 Executor pro Worker bereit.
  - Für den Worker-Typ `G.2X` stellt jeder Worker 8 vCPUs, 32 GB Speicher und 128 GB Festplattenplatz sowie 1 Executor pro Worker bereit.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType`, die zugewiesen werden, wenn diese Aufgabe ausgeführt wird.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Zeitlimit für eine Aufgabenausführung dieser Transformation in wenigen Minuten. Dies ist die maximale Zeit, für die eine Aufgabenausführung dieser Transformation Ressourcen verbrauchen kann, bevor sie beendet wird und in den Status `TIMEOUT` übergeht. Der Standardwert beträgt 2 880 Minuten (48 Stunden).

- `MaxRetries` – Zahl (Ganzzahl).

Die maximale Anzahl an Versuchen, eine Aufgabe für diese Transformation erneut zu versuchen, nachdem eine Aufgabenausführung fehlschlägt.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung für die Transformation, die aktualisiert wurde.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `AccessDeniedException`

MLTransform Aktion löschen (Python: delete_ml_transform)

Löscht eine Transformation für maschinelles Lernen. AWS Glue Machine Learning-Transformationen sind eine besondere Art von Transformation, die mithilfe von Machine Learning die Details der auszuführenden Transformation erlernen, indem sie von Beispielen lernen, die von Menschen bereitgestellt werden. Diese Transformationen werden dann von gespeichert. AWS Glue
Wenn Sie eine Transformationsdatei nicht mehr benötigen, können Sie sie löschen, indem Sie `DeleteMLTransforms` aufrufen. Alle AWS Glue Jobs, die immer noch auf die gelöschte Transformation verweisen, sind jedoch nicht mehr erfolgreich.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der zu löschenden Transformation.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Transformation, die gelöscht wurde.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

MLTransform Aktion abrufen (Python: get_ml_transform)

Ruft ein Transformationsartefakt für AWS Glue maschinelles Lernen und alle zugehörigen Metadaten ab. Machine Learning-Transformationen sind eine besondere Art von Transformation, die mithilfe von Machine Learning die Details der auszuführenden Transformation erlernen, indem sie von Beispielen lernen, die von Menschen bereitgestellt werden. Diese Transformationen werden dann von gespeichert. AWS Glue Sie können ihre Metadaten abrufen, indem Sie `GetMLTransform` aufrufen.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Transformation, der zu dem Zeitpunkt generiert wurde, als die Transformation erstellt wurde.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Transformation, der zu dem Zeitpunkt generiert wurde, als die Transformation erstellt wurde.

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der eindeutige Name, den die Transformation erhielt, als sie erstellt wurde.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Transformation.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `NOT_READY` | `READY` | `DELETING`).

  Der letzte bekannte Status der Transformation (um anzugeben, ob sie verwendet werden kann). Möglich sind „NOT_READY“, „READY“ oder „DELETING“.

- `CreatedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Transformierung erstellt wurde.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Transformation zuletzt geändert wurde.

- `InputRecordTables` – Ein Array mit GlueTable-Objekten, nicht mehr als 10 Strukturen.

  Eine Liste der von der Transformation verwendeten AWS Glue Tabellendefinitionen.

- `Parameters` – Ein TransformParameters-Objekt.

  Die Konfigurationsparameter, die für den verwendeten Algorithmus spezifisch sind.

- `EvaluationMetrics` – Ein EvaluationMetrics-Objekt.

  Die neueste Auswertungsmetriken.

- `LabelCount` – Zahl (Ganzzahl).

  Die Anzahl der für diese Transformation verfügbaren Labels.

- `Schema` – Ein Array mit SchemaColumn-Objekten, nicht mehr als 100 Strukturen.

  Das `Map<Column, Type>`-Objekt, durch das das Schema repräsentiert wird, das von dieser Transformation akzeptiert wird. Hat eine Obergrenze von 100 Spalten.

- `Role` – UTF-8-Zeichenfolge.

  Der Name oder Amazon-Ressourcenname (ARN) der IAM-Rolle mit den erforderlichen Berechtigungen.

- `GlueVersion` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #47.

  Dieser Wert bestimmt, mit welcher Version AWS Glue dieser Transformation für maschinelles Lernen kompatibel ist. Glue 1.0 wird für die meisten Kunden empfohlen. Wenn der Wert nicht festgelegt ist, wird die Glue-Kompatibilität standardmäßig auf Glue 0.9 gesetzt. Weitere Informationen finden Sie unter AWS Glue -Versionen im Entwicklerhandbuch.

- `MaxCapacity` – Nummer (doppelt).

  Die Anzahl der AWS Glue Datenverarbeitungseinheiten (DPUs), die Aufgabenausführungen für diese Transformation zugewiesen sind. Sie können 2 bis 100 zuweisen. DPUs Die Standardeinstellung ist 10. Eine DPU ist eine relative Maßeinheit für die Rechenleistung, die sich aus 4 V CPUs Rechenkapazität und 16 GB Arbeitsspeicher zusammensetzt. Weitere Informationen finden Sie in der AWS Glue Preisliste.

  Wenn das Feld `WorkerType` auf einen anderen Wert als `Standard` gesetzt ist, wird das Feld `MaxCapacity` automatisch eingestellt und wird schreibgeschützt.

- `WorkerType` – UTF-8-Zeichenfolge (zulässige Werte: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Der Typ des vordefinierten Workers, der zugeordnet wird, wenn diese Aufgabe ausgeführt wird. Akzeptiert den Wert Standard, G.1X oder G.2X.

  - Für den Worker-Typ `Standard` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 50 GB Festplattenplatz sowie 2 Executors pro Worker bereit.
  - Für den Worker-Typ `G.1X` stellt jeder Worker 4 vCPUs, 16 GB Arbeitsspeicher und 64 GB Festplattenplatz sowie 1 Executor pro Worker bereit.
  - Für den Worker-Typ `G.2X` stellt jeder Worker 8 vCPUs, 32 GB Speicher und 128 GB Festplattenplatz sowie 1 Executor pro Worker bereit.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der Worker eines definierten `workerType`, die zugewiesen werden, wenn diese Aufgabe ausgeführt wird.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Zeitlimit für eine Aufgabenausführung dieser Transformation in wenigen Minuten. Dies ist die maximale Zeit, für die eine Aufgabenausführung dieser Transformation Ressourcen verbrauchen

kann, bevor sie beendet wird und in den Status `TIMEOUT` übergeht. Der Standardwert beträgt 2 880 Minuten (48 Stunden).

- `MaxRetries` – Zahl (Ganzzahl).

  Die maximale Anzahl an Versuchen, eine Aufgabe für diese Transformation erneut zu versuchen, nachdem eine Aufgabenausführung fehlschlägt.

- `TransformEncryption` – Ein TransformEncryption-Objekt.

  Die encryption-at-rest Einstellungen der Transformation, die für den Zugriff auf Benutzerdaten gelten. Machine-Learning-Transformationen können mithilfe von KMS auf in Amazon S3 verschlüsselte Benutzerdaten zugreifen.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

MLTransforms Aktion abrufen (Python: get_ml_transforms)

Ruft eine sortierbare, filterbare Liste vorhandener Transformationen für maschinelles Lernen ab. AWS Glue Machine Learning-Transformationen sind eine besondere Art von Transformation, die mithilfe von Machine Learning die Details der auszuführenden Transformation erlernen, indem sie von Beispielen lernen, die von Menschen bereitgestellt werden. Diese Transformationen werden dann von gespeichert AWS Glue, und Sie können ihre Metadaten abrufen, indem Sie aufrufen. `GetMLTransforms`

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.

- `Filter` – Ein TransformFilterCriteria-Objekt.

Die Filterkritieren der Transformation.

- `Sort` – Ein TransformSortCriteria-Objekt.

  Die Sortierkriterien.

Antwort

- `Transforms` – Erforderlich: Ein Array mit MLTransform-Objekten.

  Eine Liste der Machine Learning-Transformationen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

MLTransforms Aktion auflisten (Python: list_ml_transforms)

Ruft eine sortierbare, filterbare Liste vorhandener Transformationen für AWS Glue maschinelles Lernen in diesem AWS Konto oder der Ressourcen mit dem angegebenen Tag ab. Diese Operation akzeptiert das optionale `Tags`-Feld, das Sie als Filter der Antworten verwenden können, sodass markierte Ressourcen als Gruppe abgerufen werden können. Wenn Sie die Tag-Filterung verwenden, werden nur Ressourcen mit den Tags abgerufen.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn dies eine Fortsetzungsanforderung ist.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Größe der auszugebenden Liste.

- `Filter` – Ein TransformFilterCriteria-Objekt.

  Ein `TransformFilterCriteria` zum Filtern der Machine Learning-Transformationen.

- `Sort` – Ein TransformSortCriteria-Objekt.

  A `TransformSortCriteria` zum Sortieren der Machine Learning-Transformationen.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Gibt an, das nur diese markierten Ressourcen zurückgegeben werden sollen.

Antwort

- `TransformIds` – Erforderlich: Ein Array von UTF-8-Zeichenketten.

  Die IDs (Kennungen) aller Machine Learning-Transformationen in dem Konto oder die Machine Learning-Transformationen mit den angegebenen Tags.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Fortsetzungs-Token, wenn die zurückgegebene Liste die letzte verfügbare Metrik nicht enthält.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

MLEvaluationTaskRun Aktion starten (Python: start_ml_evaluation_task_run)

Startet eine Aufgabe zur Schätzung der Qualität der Transformation.

Wenn Sie Labelsätze als Wahrheitsbeispiele angeben, verwendet AWS Glue maschinelles Lernen einige dieser Beispiele, um daraus zu lernen. Die restlichen Labels werden zum Testen der Schätzungsqualität verwendet.

Gibt eine eindeutige Kennung für die Ausführung zurück. Sie können `GetMLTaskRun` aufrufen, um weitere Informationen über die Statistiken des `EvaluationTaskRun` zu erhalten.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.

Antwort

- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConcurrentRunsExceededException`
- `MLTransformNotReadyException`

MLLabelingSetGenerationTaskRun Aktion starten (Python: start_ml_labeling_set_generation_task_run)

Startet den aktiven Learning-Workflow für Ihre Machine Learning-Transformation, um die Qualität der Transformation durch Generieren von Label-Sätzen und Hinzufügen von Labels zu verbessern.

Nach Abschluss von `StartMLLabelingSetGenerationTaskRun` hat AWS Glue einen Labeling-Satz oder einen Satz von Fragen zur Beantwortung durch Menschen generiert.

Im Falle der Transformation `FindMatches` stellen sich in etwa folgende Fragen: „Wie lassen sich diese Zeilen richtig als Gruppen zusammen gruppieren, die ausschließlich aus übereinstimmenden Datensätze bestehen?“

Nachdem der Labeling-Prozess abgeschlossen wurde, können Sie Ihre Labels mit einem Aufruf in die `StartImportLabelsTaskRun` hochladen. Nachdem die `StartImportLabelsTaskRun` abgeschlossen wurde, verwenden alle zukünftigen Ausführungen der Machine Learning-Transformation die neuen und verbesserten Labels und führen eine Transformation mit höherer Qualität durch.

Hinweis: Die Rolle, die verwendet wird, um den generierten Labelsatz in die zu schreiben, `OutputS3Path` ist die Rolle, die der in der `CreateMLTransform` API angegebenen Machine Learning Learning-Transformation zugeordnet ist.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `OutputS3Path` – Erforderlich: UTF-8-Zeichenfolge.

  Der Amazon Simple Storage Service (Amazon S3)-Pfad, in dem Sie das Labeling-Set generieren.

Antwort

- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Ausführungskennung, die dieser Aufgabe zugeordnet ist.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

- `InternalServiceException`
- `ConcurrentRunsExceededException`

Aktion „MLTaskAusführen abrufen“ (Python: get_ml_task_run)

Ruft Details für eine bestimmte Aufgabenausführung für eine Machine Learning-Transformation ab. Aufgabenausführungen für maschinelles Lernen sind asynchrone Aufgaben, die in Ihrem Namen als Teil verschiedener Workflows für maschinelles Lernen AWS Glue ausgeführt werden. Sie können die Statistiken jeder beliebigen Aufgabenausführung überprüfen, indem Sie die `GetMLTaskRun` mit der `TaskRunID` und der `TransformID` seiner übergeordneten Transformation aufrufen.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `TaskRunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Aufgabenausführung.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Aufgabenausführung.
- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Aufgabenausführung.
- `LogGroupName` – UTF-8-Zeichenfolge.

Die Namen der Protokollgruppen, die der Aufgabenausführung zugeordnet sind.

- `Properties` – Ein TaskRunProperties-Objekt.

  Die Liste der Eigenschaften, die der Aufgabenausführung zugeordnet sind.

- `ErrorString` – UTF-8-Zeichenfolge.

  Die Fehlerzeichenfolgen, die der Aufgabenausführung zugeordnet sind.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Aufgabenausführung gestartet wurde.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Aufgabe zuletzt geändert wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Aufgabenausführung abgeschlossen wurde.

- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Aufgabenausführung Ressourcen verbraucht wurden.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Aktion „Get MLTask Runs“ (Python: get_ml_task_runs)

Ruft eine Liste der Ausführungen für eine Machine Learning-Transformation ab. Aufgabenausführungen für maschinelles Lernen sind asynchrone Aufgaben, die in Ihrem Namen als Teil verschiedener Workflows für maschinelles Lernen AWS Glue ausgeführt werden. Sie erhalten eine sortierbare, filterbare Liste der Machine Learning-Aufgabenausführungen, indem Sie `GetMLTaskRuns` mit der `TransformID` ihrer übergeordneten Transformation und anderen optionalen Parametern wie in diesem Abschnitt dokumentiert aufrufen.

Diese Operation gibt eine Liste der historischen Ausführungen zurück und muss paginiert werden.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Token für die Paginierung der Ergebnisse. Der Standardwert ist leer.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.
- `Filter` – Ein TaskRunFilterCriteria-Objekt.

  Die Filterkriterien in der `TaskRunFilterCriteria`-Struktur für die Aufgabenausführung.
- `Sort` – Ein TaskRunSortCriteria-Objekt.

  Die Sortierungskriterien in der `TaskRunSortCriteria`-Struktur für die Aufgabenausführung.

Antwort

- `TaskRuns` – Ein Array mit TaskRun-Objekten.

  Eine Liste von Aufgabenausführungen, die der Transformation zugeordnet sind.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Aktion „MLTaskAusführen“ abbrechen (Python: cancel_ml_task_run)

Storniert (stoppt) eine Aufgabenausführung. Aufgabenausführungen für maschinelles Lernen sind asynchrone Aufgaben, die in Ihrem Namen als Teil verschiedener Workflows für maschinelles Lernen AWS Glue ausgeführt werden. Sie können eine Machine Learning-Aufgabenausführung jederzeit stornieren, indem Sie `CancelMLTaskRun` mit der `TransformID` der übergeordneten Transformation der Aufgabenausführung und und die `TaskRunId` der Aufgabenausführung aufrufen.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `TaskRunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Aufgabenausführung.

Antwort

- `TransformId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Aufgabenausführung.
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`

StartExportLabelsTaskRun Aktion (Python: start_export_labels_task_run)

Startet eine asynchrone Aufgabe zum Exportieren aller gelabelten Daten für eine bestimmte Transformation. Diese Aufgabe ist der einzige Label-bezogene API-Aufruf, der nicht Teil des typischen aktiven Learning-Workflows ist. In der Regel verwenden Sie `StartExportLabelsTaskRun`, wenn Sie mit allen Ihren vorhandenen Labels gleichzeitig zusammenarbeiten möchten, wie beispielsweise, wenn Sie Labels entfernen oder ändern möchten, die zuvor als realistische Daten übertragen wurden. Diese API-Operation akzeptiert die `TransformId`, deren Labels Sie exportieren möchten, und einen Amazon Simple Storage Service (Amazon S3)-Pfad, in den die Labels exportiert werden sollen. Die Operation gibt eine `TaskRunId` zurück. Sie können den Status Ihrer Aufgabenausführung durch Aufruf der `GetMLTaskRun`-API überprüfen.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.
- `OutputS3Path` – Erforderlich: UTF-8-Zeichenfolge.

  Der Amazon S3-Pfad, in den Sie die Labels exportieren.

Antwort

- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Aufgabenausführung.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`

StartImportLabelsTaskRun Aktion (Python: start_import_labels_task_run)

Ermöglicht Ihnen die Angabe zusätzlicher Labels (realistischer Beispiele), anhand denen die Machine-Learning-Transformation lernen und ihre Qualität verbessern kann. Diese API-Operation wird allgemein als Teil des aktiven Learning-Workflows verwendet, der mit dem `StartMLLabelingSetGenerationTaskRun`-Aufruf beginnt und letztendlich zur Verbesserung der Qualität Ihrer Machine Learning-Transformation führt.

Bei Abschluss von `StartMLLabelingSetGenerationTaskRun` hat AWS Glue Machine Learning eine Reihe von Fragen zur Beantwortung durch Menschen generiert. (Die Beantwortung dieser Fragen wird häufig als 'Labeling' in den Machine Learning-Workflows bezeichnet). Im Falle der Transformation `FindMatches` stellen sich in etwa folgende Fragen: „Wie lassen sich diese Zeilen richtig als Gruppen zusammen gruppieren, die ausschließlich aus übereinstimmenden Datensätze bestehen?“ Nachdem der Kennzeichnungsvorgang abgeschlossen ist, laden Benutzer ihre Daten mit einem Aufruf an hoch. answers/labels `StartImportLabelsTaskRun` Nach Abschluss von `StartImportLabelsTaskRun` verwenden alle zukünftigen Ausführungen der Machine Learning-Transformation die neuen und verbesserten Labels und führen eine Transformation mit höherer Qualität durch.

Standardmäßig lernt `StartMLLabelingSetGenerationTaskRun` kontinuierlich von allen Labels, die Sie hochladen, und kombiniert diese miteinander, sofern Sie `Replace` nicht auf „true“ setzen. Wenn Sie `Replace` auf „true“ setzen, löscht und vergisst `StartImportLabelsTaskRun` alle zuvor hochgeladenen Labels und lernt nur von dem exakten Satz, den Sie hochladen. Das Ersetzen von Labels kann hilfreich sein, wenn Sie feststellen, dass Sie zuvor inkorrekte Labels hochgeladen haben und Sie der Ansicht sind, dass sie eine negative Auswirkung auf die Qualität der Transformation haben.

Sie können den Status Ihrer Aufgabe durch Aufrufen der Operation `GetMLTaskRun` überprüfen.

Anforderung

- `TransformId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Kennung der Machine Learning-Transformation.

- `InputS3Path` – Erforderlich: UTF-8-Zeichenfolge.

  Der Amazon Simple Storage Service (Amazon S3)-Pfad, aus dem Sie die Labels importieren.
- `ReplaceAllLabels` – Boolesch.

  Gibt an, ob Ihre vorhandenen Bezeichnungen überschrieben werden sollen.

Antwort

- `TaskRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Kennung für die Aufgabenausführung.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`

Data-Quality-API

Die Data-Quality-API beschreibt die Data-Quality-Datentypen und umfasst die API zum Erstellen, Löschen oder Aktualisieren von Datenqualitätsregelsätzen, -Ausführungen und -Bewertungen.

Datentypen

- DataSource Struktur
- DataQualityRulesetListDetails Struktur
- DataQualityTargetTable Struktur
- DataQualityRulesetEvaluationRunDescription Struktur
- DataQualityRulesetEvaluationRunFilter Struktur
- DataQualityEvaluationRunAdditionalRunOptions Struktur

- DataQualityRuleRecommendationRunDescription Struktur
- DataQualityRuleRecommendationRunFilter Struktur
- DataQualityResult Struktur
- DataQualityAnalyzerResult Struktur
- DataQualityObservation Struktur
- MetricBasedObservation Struktur
- DataQualityMetricValues Struktur
- DataQualityRuleResult Struktur
- DataQualityResultDescription Struktur
- DataQualityResultFilterCriteria Struktur
- DataQualityRulesetFilterCriteria Struktur
- DataQualityAggregatedMetrics Struktur
- StatisticAnnotation Struktur
- TimestampedInclusionAnnotation Struktur
- AnnotationError Struktur
- DatapointInclusionAnnotation Struktur
- StatisticSummaryList Liste
- StatisticSummary Struktur
- RunIdentifier Struktur
- StatisticModelResult Struktur

DataSource Struktur

Eine Datenquelle (eine AWS Glue Tabelle), für die Sie Datenqualitätsergebnisse wünschen.

Felder

- `GlueTable` – Erforderlich: Ein GlueTable-Objekt.

  Eine AWS Glue Tabelle.

DataQualityRulesetListDetails Struktur

Beschreibt einen Datenqualitätsregelsatz, der von `GetDataQualityRuleset` zurückgegeben wurde.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes für die Datenqualität.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes für die Datenqualität.

- `CreatedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der der Datenqualitätsregelsatz erstellt wurde.

- `LastModifiedOn` – Zeitstempel.

  Das Datum und die Uhrzeit der letzten Änderung des Regelsatzes für die Datenqualität.

- `TargetTable` – Ein DataQualityTargetTable-Objekt.

  Ein Objekt, das eine AWS Glue Tabelle darstellt.

- `RecommendationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Wenn ein Regelsatz aus einer Empfehlungsausführung erstellt wurde, wird diese Ausführungs-ID generiert, um die beiden miteinander zu verknüpfen.

- `RuleCount` – Zahl (Ganzzahl).

  Die Anzahl der Regeln im Regelsatz.

DataQualityTargetTable Struktur

Ein Objekt, das eine AWS Glue Tabelle darstellt.

Felder

- `TableName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der AWS Glue Tabelle.

- `DatabaseName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenbank, in der die AWS Glue Tabelle existiert.

- `CatalogId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Katalog-ID, in der die AWS Glue Tabelle existiert.

DataQualityRulesetEvaluationRunDescription Struktur

Beschreibt das Ausführungsergebnis zur Auswertung des Datenqualitätsregelsatzes.

Felder

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Ausführung gestartet wurde.

- `DataSource` – Ein DataSource-Objekt.

  Die dem Lauf zugeordnete Datenquelle (eine AWS Glue Tabelle).

DataQualityRulesetEvaluationRunFilter Struktur

Die Filterkriterien.

Felder

- `DataSource` – Erforderlich: Ein DataSource-Objekt.

  Filter basierend auf einer Datenquelle (einer AWS Glue Tabelle), die dem Lauf zugeordnet ist.
- `StartedBefore` – Zeitstempel.

  Filtert Ergebnisse nach Ausführungen, die vor diesem Zeitpunkt begonnen haben.
- `StartedAfter` – Zeitstempel.

  Filtert Ergebnisse nach Ausführungen, die nach diesem Zeitpunkt begonnen haben.

DataQualityEvaluationRunAdditionalRunOptions Struktur

Zusätzliche Ausführungsoptionen, die Sie für eine Ausführung der Auswertung angeben können.

Felder

- `CloudWatchMetricsEnabled` – Boolesch.

  Ob CloudWatch Metriken aktiviert werden sollen oder nicht.
- `ResultsS3Prefix` – UTF-8-Zeichenfolge.

  Präfix für Amazon S3 zum Speichern von Ergebnissen.
- `CompositeRuleEvaluationMethod` – UTF-8-Zeichenfolge (zulässige Werte: `COLUMN` | `ROW`).

  Legen Sie die Bewertungsmethode für zusammengesetzte Regeln im Regelsatz auf ROW/COLUMN fest

DataQualityRuleRecommendationRunDescription Struktur

Beschreibt das Ergebnis einer Empfehlungsausführung einer Datenqualitätsregel.

Felder

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung gestartet wurde.

- `DataSource` – Ein DataSource-Objekt.

  Die Datenquelle (AWS Glue Tabelle), die dem Empfehlungslauf zugeordnet ist.

DataQualityRuleRecommendationRunFilter Struktur

Ein Filter zum Auflisten von Empfehlungsausführungen zu Datenqualität.

Felder

- `DataSource` – Erforderlich: Ein DataSource-Objekt.

  Filter basierend auf einer angegebenen Datenquelle (AWS Glue Tabelle).

- `StartedBefore` – Zeitstempel.

  Filtert basierend auf der Zeit für Ergebnisse, die vor der angegebenen Zeit gestartet wurden.

- `StartedAfter` – Zeitstempel.

  Filtert basierend auf der Zeit für Ergebnisse, die nach der angegebenen Zeit gestartet wurden.

DataQualityResult Struktur

Beschreibt ein Datenqualitätsergebnis.

Felder

- `ResultId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ergebnis-ID für das Datenqualitätsergebnis.
- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID für das Datenqualitätsergebnis.
- `Score` – Zahl (Double), nicht mehr als 1,0.

  Ein aggregierter Datenqualitätswert. Stellt das Verhältnis der Regeln dar, die an die Gesamtzahl der Regeln übergeben wurden.
- `DataSource` – Ein DataSource-Objekt.

  Die Tabelle, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.
- `RulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes, der dem Datenqualitätsergebnis zugeordnet ist.
- `EvaluationContext` – UTF-8-Zeichenfolge.

  Im Kontext eines Jobs in AWS Glue Studio wird in der Regel jedem Knoten auf der Arbeitsfläche ein Name zugewiesen, und Datenqualitätsknoten haben Namen. Bei mehreren Knoten kann das `evaluationContext` die Knoten unterscheiden.
- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Datenqualitätsausführung gestartet wurde.
- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Datenqualitätsausführung abgeschlossen wurde.
- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Auftragsname, der dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.
- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die Auftragsausführungs-ID, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `RulesetEvaluationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Ausführungs-ID für die Regelsatzauswertung für dieses Datenqualitätsergebnis.

- `RuleResults` – Ein Array mit DataQualityRuleResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityRuleResult`-Objekten, die die Ergebnisse für jede Regel darstellen.

- `AnalyzerResults` – Ein Array mit DataQualityAnalyzerResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityAnalyzerResult`-Objekten, die die Ergebnisse für jede Analysator darstellen.

- `Observations` – Ein Array mit DataQualityObservation-Objekten, nicht mehr als 50 Strukturen.

  Eine Liste von `DataQualityObservation`-Objekten, die die Beobachtungen darstellen, die nach der Auswertung der Regeln und Analysatoren generiert wurden.

- `AggregatedMetrics` – Ein DataQualityAggregatedMetrics-Objekt.

  Eine Zusammenfassung der `DataQualityAggregatedMetrics` Objekte mit der Gesamtzahl der verarbeiteten Zeilen und Regeln, einschließlich ihrer pass/fail Statistiken, die auf Ergebnissen auf Zeilenebene basieren.

DataQualityAnalyzerResult Struktur

Beschreibt das Ergebnis der Bewertung eines Datenqualitätsanalysators.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Datenqualitätsanalysators.

- `Description` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Datenqualitätsanalysators.

- `EvaluationMessage` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Bewertungsmeldung.

- `EvaluatedMetrics` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine Zahl (doppelt).

  Eine Übersicht der Messwerte im Zusammenhang mit der Bewertung des Analysators.

DataQualityObservation Struktur

Beschreibt die Beobachtung, die nach der Auswertung der Regeln und Analysatoren generiert wurde.

Felder

- `Description` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Beobachtung der Datenqualität.

- `MetricBasedObservation` – Ein MetricBasedObservation-Objekt.

  Ein Objekt vom Typ, das die Beobachtung `MetricBasedObservation` darstellt, die auf bewerteten Datenqualitätsmetriken basiert.

MetricBasedObservation Struktur

Beschreibt die metrikbasierte Beobachtung, die auf der Grundlage ausgewerteter Datenqualitätskennzahlen generiert wurde.

Felder

- `MetricName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenqualitätsmetrik, die zur Generierung der Beobachtung verwendet wurde.

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.

- `MetricValues` – Ein DataQualityMetricValues-Objekt.

  Ein Objekt vom Typ, das die Analyse des Metrikwerts für die Datenqualität `DataQualityMetricValues` darstellt.

- `NewRules` – Ein UTF-8-Zeichenfolgen-Array.

  Eine Liste neuer Datenqualitätsregeln, die im Rahmen der Beobachtung auf der Grundlage des Metrikwerts für die Datenqualität generiert wurden.

DataQualityMetricValues Struktur

Beschreibt den Wert der Datenqualitätsmetrik anhand der Analyse historischer Daten.

Felder

- `ActualValue` – Nummer (doppelt).

  Der tatsächliche Wert der Datenqualitätsmetrik.

- `ExpectedValue` – Nummer (doppelt).

  Der erwartete Wert der Datenqualitätsmetrik gemäß der Analyse historischer Daten.

- `LowerLimit` – Nummer (doppelt).

  Die Untergrenze des Werts der Datenqualitätsmetrik gemäß der Analyse historischer Daten.

- `UpperLimit` – Nummer (doppelt).

  Die Obergrenze des Metrikwerts für die Datenqualität gemäß der Analyse historischer Daten.

DataQualityRuleResult Struktur

Beschreibt das Ergebnis der Auswertung einer Datenqualitätsregel.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Datenqualitätsregel.

- `Description` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung der Datenqualitätsregel.

- `EvaluationMessage` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Bewertungsmeldung.

- `Result` – UTF-8-Zeichenfolge (zulässige Werte: `PASS` | `FAIL` | `ERROR`).

  Ein Status „Bestanden“ oder „Nicht bestanden“ für die Regel.

- `EvaluatedMetrics` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine Zahl (doppelt).

  Eine Zuordnung der Metriken, die der Auswertung der Regel zugewiesen sind.

- `EvaluatedRule` – UTF-8-Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die ausgewertete Regel.

- `RuleMetrics` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine Zahl (doppelt).

  Eine Übersicht mit Metriken im Zusammenhang mit der Auswertung der Regel auf der Grundlage von Ergebnissen auf Zeilenebene.

DataQualityResultDescription Struktur

Beschreibt ein Datenqualitätsergebnis.

Felder

- `ResultId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Ergebnis-ID für dieses Datenqualitätsergebnis.
- `DataSource` – Ein DataSource-Objekt.

  Der dem Datenqualitätsergebnis zugeordnete Tabellenname.
- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der dem Datenqualitätsergebnis zugeordnete Auftragsname.
- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftragsausführungs-ID, die dem Datenqualitätsergebnis zugeordnet ist.
- `StartedOn` – Zeitstempel.

  Die Uhrzeit, zu der die Ausführung für dieses Datenqualitätsergebnis gestartet wurde.

DataQualityResultFilterCriteria Struktur

Kriterien, die für die Rückgabe von Datenqualitätsergebnissen verwendet werden.

Felder

- `DataSource` – Ein DataSource-Objekt.

  Filtert Ergebnisse nach der angegebenen Datenquelle. Zum Beispiel das Abrufen aller Ergebnisse für eine AWS Glue Tabelle.
- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Filtert Ergebnisse nach dem angegebenen Auftragsnamen.

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Filtert Ergebnisse nach der angegebenen Auftragsausführung-ID.
- `StartedAfter` – Zeitstempel.

  Filtert Ergebnisse nach Ausführungen, die nach diesem Zeitpunkt begonnen haben.
- `StartedBefore` – Zeitstempel.

  Filtert Ergebnisse nach Ausführungen, die vor diesem Zeitpunkt begonnen haben.

DataQualityRulesetFilterCriteria Struktur

Die Kriterien, die zum Filtern von Datenqualitätsregelsätzen verwendet werden.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Regelsatz-Filterkriterien.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Die Beschreibung der Regelsatz-Filterkriterien.
- `CreatedBefore` – Zeitstempel.

  Filtert nach Regelsätzen, die vor diesem Datum erstellt wurden.
- `CreatedAfter` – Zeitstempel.

  Filtert nach Regelsätzen, die nach diesem Datum erstellt wurden.
- `LastModifiedBefore` – Zeitstempel.

  Filtert nach Regelsätzen, die zuletzt vor diesem Datum geändert wurden.
- `LastModifiedAfter` – Zeitstempel.

  Filter nach Regelsätzen, die zuletzt nach diesem Datum geändert wurden.
- `TargetTable` – Ein DataQualityTargetTable-Objekt.

Der Name und der Datenbankname der Zieltabelle.

DataQualityAggregatedMetrics Struktur

Eine Zusammenfassung der Metriken, die die Gesamtzahl der verarbeiteten Zeilen und Regeln zeigt, einschließlich ihrer pass/fail Statistiken, die auf Ergebnissen auf Zeilenebene basieren.

Felder

- `TotalRowsProcessed` – Nummer (doppelt).

  Die Gesamtzahl der Zeilen, die während der Bewertung der Datenqualität verarbeitet wurden.
- `TotalRowsPassed` – Nummer (doppelt).

  Die Gesamtzahl der Zeilen, die alle geltenden Datenqualitätsregeln erfüllt haben.
- `TotalRowsFailed` – Nummer (doppelt).

  Die Gesamtzahl der Zeilen, die eine oder mehrere Datenqualitätsregeln nicht bestanden haben.
- `TotalRulesProcessed` – Nummer (doppelt).

  Die Gesamtzahl der Datenqualitätsregeln, die bewertet wurden.
- `TotalRulesPassed` – Nummer (doppelt).

  Die Gesamtzahl der Datenqualitätsregeln, die ihre Bewertungskriterien erfüllt haben.
- `TotalRulesFailed` – Nummer (doppelt).

  Die Gesamtzahl der Datenqualitätsregeln, die ihre Bewertungskriterien nicht erfüllt haben.

StatisticAnnotation Struktur

Eine statistische Anmerkung.

Felder

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID.

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.

- `StatisticRecordedOn` – Zeitstempel.

  Der Zeitstempel, zu dem die kommentierte Statistik aufgezeichnet wurde.

- `InclusionAnnotation` – Ein TimestampedInclusionAnnotation-Objekt.

  Die Inklusionsanmerkung, die auf die Statistik angewendet wurde.

TimestampedInclusionAnnotation Struktur

Eine Anmerkung zur Inklusion mit einem Zeitstempel.

Felder

- `Value` – UTF-8-Zeichenfolge (zulässige Werte: `INCLUDE` | `EXCLUDE`).

  Der Wert der Inklusionsanmerkung.

- `LastModifiedOn` – Zeitstempel.

  Der Zeitstempel, zu dem die Inklusionsanmerkung zuletzt geändert wurde.

AnnotationError Struktur

Eine fehlgeschlagene Anmerkung.

Felder

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID für die fehlgeschlagene Anmerkung.

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID für die fehlgeschlagene Anmerkung.

- `FailureReason` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Der Grund, warum die Anmerkung fehlgeschlagen ist.

DatapointInclusionAnnotation Struktur

Eine Anmerkung zur Inklusion.

Felder

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Datenqualitätsprofils, zu dem die Statistik gehört.
- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.
- `InclusionAnnotation` – UTF-8-Zeichenfolge (zulässige Werte: `INCLUDE` | `EXCLUDE`).

  Der Wert der Inklusionsanmerkung, der auf die Statistik angewendet werden soll.

StatisticSummaryList Liste

Eine Liste von `StatisticSummary`.

Ein Array von StatisticSummary-Objekten.

Eine Liste von `StatisticSummary`.

StatisticSummary Struktur

Zusammenfassende Informationen zu einer Statistik.

Felder

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die Statistik-ID.

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID.

- `RunIdentifier` – Ein RunIdentifier-Objekt.

  Die Lauf-ID

- `StatisticName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Custom string pattern #16.

  Der Name der Statistik.

- `DoubleValue` – Nummer (doppelt).

  Der Wert der Statistik.

- `EvaluationLevel` – UTF-8-Zeichenfolge (zulässige Werte: `Dataset="DATASET"` | `Column="COLUMN"` | `Multicolumn="MULTICOLUMN"`).

  Die Bewertungsebene der Statistik. Mögliche Werte:`Dataset`,`Column`,`Multicolumn`.

- `ColumnsReferenced` – Ein UTF-8-Zeichenfolgen-Array.

  Die Liste der Spalten, auf die in der Statistik verwiesen wird.

- `ReferencedDatasets` – Ein UTF-8-Zeichenfolgen-Array.

  Die Liste der Datensätze, auf die in der Statistik verwiesen wird.

- `StatisticProperties` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist eine Beschreibungszeichenfolge, die nicht länger als 2 048 Byte ist und mit URI address multi-line string pattern übereinstimmt.

  A`StatisticPropertiesMap`, das ein und enthält `NameString` `DescriptionString`

- `RecordedOn` – Zeitstempel.

  Der Zeitstempel, zu dem die Statistik aufgezeichnet wurde.

- `InclusionAnnotation` – Ein TimestampedInclusionAnnotation-Objekt.

  Die Inklusionsanmerkung für die Statistik.

RunIdentifier Struktur

Eine Lauf-ID.

Felder

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Run-ID.
- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Job Run-ID.

StatisticModelResult Struktur

Das Ergebnis des statistischen Modells.

Felder

- `LowerBound` – Nummer (doppelt).

  Die Untergrenze.
- `UpperBound` – Nummer (doppelt).

  Die Obergrenze.
- `PredictedValue` – Nummer (doppelt).

  Der vorhergesagte Wert.
- `ActualValue` – Nummer (doppelt).

  Der tatsächliche Wert.
- `Date` – Zeitstempel.

Das Datum.

- `InclusionAnnotation` – UTF-8-Zeichenfolge (zulässige Werte: `INCLUDE` | `EXCLUDE`).

  Die Anmerkung zur Inklusion.

Operationen

- StartDataQualityRulesetEvaluationRun Aktion (Python: start_data_quality_ruleset_evaluation_run)
- CancelDataQualityRulesetEvaluationRun Aktion (Python: cancel_data_quality_ruleset_evaluation_run)
- GetDataQualityRulesetEvaluationRun Aktion (Python: get_data_quality_ruleset_evaluation_run)
- ListDataQualityRulesetEvaluationRuns Aktion (Python: list_data_quality_ruleset_evaluation_runs)
- StartDataQualityRuleRecommendationRun Aktion (Python: start_data_quality_rule_recommendation_run)
- CancelDataQualityRuleRecommendationRun Aktion (Python: cancel_data_quality_rule_recommendation_run)
- GetDataQualityRuleRecommendationRun Aktion (Python: get_data_quality_rule_recommendation_run)
- ListDataQualityRuleRecommendationRuns Aktion (Python: list_data_quality_rule_recommendation_runs)
- GetDataQualityResult Aktion (Python: get_data_quality_result)
- BatchGetDataQualityResult Aktion (Python: batch_get_data_quality_result)
- ListDataQualityResults Aktion (Python: list_data_quality_results)
- CreateDataQualityRuleset Aktion (Python: create_data_quality_ruleset)
- DeleteDataQualityRuleset Aktion (Python: delete_data_quality_ruleset)
- GetDataQualityRuleset Aktion (Python: get_data_quality_ruleset)
- ListDataQualityRulesets Aktion (Python: list_data_quality_rulesets)
- UpdateDataQualityRuleset Aktion (Python: update_data_quality_ruleset)
- ListDataQualityStatistics Aktion (Python: list_data_quality_statistics)
- TimestampFilter Struktur
- CreateDataQualityRulesetRequest Struktur
- GetDataQualityRulesetResponse Struktur

- GetDataQualityResultResponse Struktur
- StartDataQualityRuleRecommendationRunRequest Struktur
- GetDataQualityRuleRecommendationRunResponse Struktur
- BatchPutDataQualityStatisticAnnotation Aktion (Python: batch_put_data_quality_statistic_annotation)
- GetDataQualityModel Aktion (Python: get_data_quality_model)
- GetDataQualityModelResult Aktion (Python: get_data_quality_model_result)
- ListDataQualityStatisticAnnotations Aktion (Python: list_data_quality_statistic_annotations)
- PutDataQualityProfileAnnotation Aktion (Python: put_data_quality_profile_annotation)

StartDataQualityRulesetEvaluationRun Aktion (Python: start_data_quality_ruleset_evaluation_run)

Sobald Sie eine Regelsatzdefinition haben (entweder empfohlen oder Ihre eigene), rufen Sie diese Operation auf, um den Regelsatz anhand einer Datenquelle (Tabelle) auszuwerten.AWS Glue Die Auswertung berechnet Ergebnisse, die Sie mit der `GetDataQualityResult`-API abrufen können.

Anforderung

- `DataSource` – Erforderlich: Ein DataSource-Objekt.

  Die diesem Lauf zugeordnete Datenquelle (AWS Glue Tabelle).
- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Eine IAM Rolle, die zur Verschlüsselung der Ergebnisse des Laufs bereitgestellt wird.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.
- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).
- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Wird für Idempotenz verwendet und sollte auf eine zufällige ID (z. B. eine UUID) festgelegt werden, um zu vermeiden, dass mehrere Instances der gleichen Ressource erstellt oder gestartet werden.

- `AdditionalRunOptions` – Ein DataQualityEvaluationRunAdditionalRunOptions-Objekt.

  Zusätzliche Ausführungsoptionen, die Sie für eine Ausführung der Auswertung angeben können.

- `RulesetNames` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 10 Zeichenfolgen.

  Eine Liste von Regelsatznamen.

- `AdditionalDataSources` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist ein A DataSource-Objekt.

  Eine Zuordnung von Zeichenfolgen zu zusätzlichen Datenquellen, die Sie für eine Ausführung der Auswertung angeben können.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Fehler

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRulesetEvaluationRun Aktion (Python: cancel_data_quality_ruleset_evaluation_run)

Bricht eine Ausführung ab, bei der ein Regelsatz anhand einer Datenquelle ausgewertet wird.

Anforderung

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRulesetEvaluationRun Aktion (Python: get_data_quality_ruleset_evaluation_run)

Ruft eine bestimmte Ausführung ab, bei der ein Regelsatz anhand einer Datenquelle ausgewertet wird.

Anforderung

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

- `DataSource` – Ein DataSource-Objekt.

  Die Datenquelle (eine Tabelle), die diesem Testlauf zugeordnet ist. AWS Glue

- `Role` – UTF-8-Zeichenfolge.

  Eine IAM Rolle, die zur Verschlüsselung der Ergebnisse des Rechenlaufs bereitgestellt wird.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).

- `AdditionalRunOptions` – Ein DataQualityEvaluationRunAdditionalRunOptions-Objekt.

  Zusätzliche Ausführungsoptionen, die Sie für eine Ausführung der Auswertung angeben können.

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.

- `ErrorString` – UTF-8-Zeichenfolge.

  Die Fehlerzeichenfolgen, die der Ausführung zugeordnet sind.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung gestartet wurde.

- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der letzte Zeitpunkt, an dem diese Empfehlungsausführung für die Datenqualitätsregel geändert wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung abgeschlossen wurde.

- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Ausführung Ressourcen verbraucht wurden.

- `RulesetNames` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 10 Zeichenfolgen.

  Eine Liste von Regelsatznamen für die Ausführung. Dieser Parameter akzeptiert derzeit nur einen Regelsatznamen.

- `ResultIds` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 10 Zeichenfolgen.

  Eine Ergebnisliste IDs für die Datenqualitätsergebnisse des Rechenlaufs.

- `AdditionalDataSources` – Ein Map-Array von Schlüssel-Wert-Paaren.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, und entspricht dem Single-line string pattern.

  Jeder Wert ist ein A DataSource-Objekt.

  Eine Zuordnung von Zeichenfolgen zu zusätzlichen Datenquellen, die Sie für eine Ausführung der Auswertung angeben können.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesetEvaluationRuns Aktion (Python: list_data_quality_ruleset_evaluation_runs)

Listet alle Ausführungen auf, die den Filterkriterien entsprechen, bei denen ein Regelsatz anhand einer Datenquelle ausgewertet wird.

Anforderung

- `Filter` – Ein DataQualityRulesetEvaluationRunFilter-Objekt.

  Die Filterkriterien.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.

Antwort

- `Runs` – Ein Array mit DataQualityRulesetEvaluationRunDescription-Objekten.

  Eine Liste von `DataQualityRulesetEvaluationRunDescription`-Objekten, die Ausführungen von Datenqualitätsregelsätzen darstellen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartDataQualityRuleRecommendationRun Aktion (Python: start_data_quality_rule_recommendation_run)

Startet einen Empfehlungslauf, der zum Generieren von Regeln verwendet wird, wenn Sie nicht wissen, welche Regeln Sie schreiben sollen. AWS Glue Data Quality analysiert die Daten und gibt Empfehlungen für einen möglichen Regelsatz ab. Sie können dann den Regelsatz auswerten und den generierten Regelsatz nach Ihren Wünschen ändern.

Empfehlungsausführungen werden nach 90 Tagen automatisch gelöscht.

Anforderung

Die Anforderung der Empfehlungsanfrage für die Datenqualitätsregel.

- `DataSource` – Erforderlich: Ein DataSource-Objekt.

  Die diesem Lauf zugeordnete Datenquelle (AWS Glue Tabelle).
- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Eine IAM Rolle, die zur Verschlüsselung der Ergebnisse des Laufs bereitgestellt wird.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.
- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).
- `CreatedRulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Name für den Regelsatz.
- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.
- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Wird für Idempotenz verwendet und sollte auf eine zufällige ID (z. B. eine UUID) festgelegt werden, um zu vermeiden, dass mehrere Instances der gleichen Ressource erstellt oder gestartet werden.

Antwort

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die dieser Ausführung zugeordnete eindeutige Kennung.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRuleRecommendationRun Aktion (Python: cancel_data_quality_rule_recommendation_run)

Bricht die angegebene Empfehlungsausführung ab, die zum Generieren von Regeln verwendet wurde.

Anforderung

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleRecommendationRun Aktion (Python: get_data_quality_rule_recommendation_run)

Ruft die angegebene Empfehlungsausführung ab, die zum Generieren von Regeln verwendet wurde.

Anforderung

- `RunId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.

Antwort

Die Antwort auf die Ausführung der Empfehlung für die Datenqualitätsregel.

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.
- `DataSource` – Ein DataSource-Objekt.

  Die diesem Lauf zugeordnete Datenquelle (eine AWS Glue Tabelle).
- `Role` – UTF-8-Zeichenfolge.

  Eine IAM Rolle, die zur Verschlüsselung der Ergebnisse des Laufs bereitgestellt wird.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.
- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.

- `ErrorString` – UTF-8-Zeichenfolge.

  Die Fehlerzeichenfolgen, die der Ausführung zugeordnet sind.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung gestartet wurde.

- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der letzte Zeitpunkt, an dem diese Empfehlungsausführung für die Datenqualitätsregel geändert wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung abgeschlossen wurde.

- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Ausführung Ressourcen verbraucht wurden.

- `RecommendedRuleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Wenn die Empfehlungsausführung einer Startregel abgeschlossen ist, wird ein empfohlener Regelsatz (ein Satz von Regeln) erstellt. Dieses Mitglied verfügt über diese Regeln im Format der Definitionssprache für Datenqualität (DQDL).

- `CreatedRulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes, der durch die Ausführung erstellt wurde.

- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

- `InternalServiceException`

ListDataQualityRuleRecommendationRuns Aktion (Python: list_data_quality_rule_recommendation_runs)

Listet die Empfehlungsausführungen auf, die die Filterkriterien erfüllen.

Anforderung

- `Filter` – Ein DataQualityRuleRecommendationRunFilter-Objekt.

  Die Filterkriterien.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.

Antwort

- `Runs` – Ein Array mit DataQualityRuleRecommendationRunDescription-Objekten.

  Eine Liste von `DataQualityRuleRecommendationRunDescription`-Objekten.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityResult Aktion (Python: get_data_quality_result)

Ruft das Ergebnis einer Auswertung einer Datenqualitätsregel ab.

Anforderung

- `ResultId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ergebnis-ID für das Datenqualitätsergebnis.

Antwort

Die Antwort auf das Datenqualitätsergebnis.

- `ResultId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ergebnis-ID für das Datenqualitätsergebnis.
- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID für das Datenqualitätsergebnis.
- `Score` – Zahl (Double), nicht mehr als 1,0.

  Ein aggregierter Datenqualitätswert. Stellt das Verhältnis der Regeln dar, die an die Gesamtzahl der Regeln übergeben wurden.
- `DataSource` – Ein DataSource-Objekt.

  Die Tabelle, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.
- `RulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes, der dem Datenqualitätsergebnis zugeordnet ist.
- `EvaluationContext` – UTF-8-Zeichenfolge.

  Im Kontext eines Jobs in AWS Glue Studio wird in der Regel jedem Knoten auf der Arbeitsfläche ein Name zugewiesen, und Datenqualitätsknoten haben Namen. Bei mehreren Knoten kann das `evaluationContext` die Knoten unterscheiden.
- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Ausführung für dieses Datenqualitätsergebnis gestartet wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Ausführung für dieses Datenqualitätsergebnis abgeschlossen wurde.

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Auftragsname, der dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftragsausführungs-ID, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `RulesetEvaluationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Ausführungs-ID, die der Regelsatzauswertung zugeordnet ist.

- `RuleResults` – Ein Array mit DataQualityRuleResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityRuleResult`-Objekten, die die Ergebnisse für jede Regel darstellen.

- `AnalyzerResults` – Ein Array mit DataQualityAnalyzerResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityAnalyzerResult`-Objekten, die die Ergebnisse für jede Analysator darstellen.

- `Observations` – Ein Array mit DataQualityObservation-Objekten, nicht mehr als 50 Strukturen.

  Eine Liste von `DataQualityObservation`-Objekten, die die Beobachtungen darstellen, die nach der Auswertung der Regeln und Analysatoren generiert wurden.

- `AggregatedMetrics` – Ein DataQualityAggregatedMetrics-Objekt.

  Eine Zusammenfassung der `DataQualityAggregatedMetrics` Objekte mit der Gesamtzahl der verarbeiteten Zeilen und Regeln, einschließlich ihrer pass/fail Statistiken, die auf Ergebnissen auf Zeilenebene basieren.

Fehler

- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`
- `EntityNotFoundException`

BatchGetDataQualityResult Aktion (Python: batch_get_data_quality_result)

Ruft eine Liste von Datenqualitätsergebnissen für das angegebene Ergebnis ab. IDs

Anforderung

- `ResultIds` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 100 Zeichenfolgen.

  Eine Liste mit eindeutigen Ergebnissen IDs für die Datenqualitätsergebnisse.

Antwort

- `Results` – Erforderlich: Ein Array mit DataQualityResult-Objekten.

  Eine Liste von `DataQualityResult`-Objekten, die die Datenqualitätsergebnisse darstellen.
- `ResultsNotFound` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 100 Zeichenfolgen.

  Eine Liste von Ergebnissen, IDs für die keine Ergebnisse gefunden wurden.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityResults Aktion (Python: list_data_quality_results)

Gibt alle Ausführungsergebnisse der Datenqualität für Ihr Konto zurück.

Anforderung

- `Filter` – Ein DataQualityResultFilterCriteria-Objekt.

Die Filterkriterien.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.

Antwort

- `Results` – Erforderlich: Ein Array mit DataQualityResultDescription-Objekten.

  Eine Liste von `DataQualityResultDescription`-Objekten.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

CreateDataQualityRuleset Aktion (Python: create_data_quality_ruleset)

Erstellt einen Regelsatz für die Datenqualität mit DQDL-Regeln, die auf eine angegebene Tabelle angewendet werden. AWS Glue

Sie erstellen den Regelsatz mit der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.

Anforderung

Eine Anfrage zur Erstellung eines Regelsatzes für die Datenqualität.

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Ein eindeutiger Name für den Datenqualitätsregelsatz.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes für die Datenqualität.

- `Ruleset` – Erforderlich: UTF-8-Zeichenkette, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Eine Liste von Tags, die auf den Datenqualitätsregelsatz angewendet werden.

- `TargetTable` – Ein DataQualityTargetTable-Objekt.

  Eine Zieltabelle, die dem Datenqualitätsregelsatz zugeordnet ist.

- `RecommendationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ausführungs-ID für die Empfehlungsausführung.

- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Wird für Idempotenz verwendet und sollte auf eine zufällige ID (z. B. eine UUID) festgelegt werden, um zu vermeiden, dass mehrere Instances der gleichen Ressource erstellt oder gestartet werden.

Antwort

- Name – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein eindeutiger Name für den Datenqualitätsregelsatz.

Fehler

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

DeleteDataQualityRuleset Aktion (Python: delete_data_quality_ruleset)

Löscht einen Datenqualitätsregelsatz.

Anforderung

- Name – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein eindeutiger Name für den Datenqualitätsregelsatz.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleset Aktion (Python: get_data_quality_ruleset)

Gibt einen vorhandenen Regelsatz nach Kennung oder Name zurück.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes.

Antwort

Gibt die Antwort auf den Regelsatz zur Datenqualität zurück.

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes.
- `Ruleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.
- `TargetTable` – Ein DataQualityTargetTable-Objekt.

  Der Name und der Datenbankname der Zieltabelle.
- `CreatedOn` – Zeitstempel.

  Ein Zeitstempel. Die Uhrzeit und das Datum, an dem dieser Datenqualitätsregelsatz erstellt wurde.
- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der letzte Zeitpunkt, an dem dieser Datenqualitätsregelsatz geändert wurde.
- `RecommendationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Wenn ein Regelsatz aus einer Empfehlungsausführung erstellt wurde, wird diese Ausführungs-ID generiert, um die beiden miteinander zu verknüpfen.

- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesets Aktion (Python: list_data_quality_rulesets)

Gibt eine paginierte Liste von Regelsätzen für die angegebene Tabellenliste zurück. AWS Glue

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.

- `Filter` – Ein DataQualityRulesetFilterCriteria-Objekt.

  Die Filterkriterien.

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

Eine Liste mit Tags für Schlüsselwertpaare.

Antwort

- `Rulesets` – Ein Array mit DataQualityRulesetListDetails-Objekten.

  Eine paginierte Liste von Regelsätzen für die angegebene Tabellenliste. AWS Glue
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

UpdateDataQualityRuleset Aktion (Python: update_data_quality_ruleset)

Aktualisiert den angegebenen Datenqualitätsregelsatz.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes für die Datenqualität.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes.
- `Ruleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im Entwicklerhandbuch. AWS Glue

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes für die Datenqualität.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes.
- `Ruleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.

Fehler

- `EntityNotFoundException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

ListDataQualityStatistics Aktion (Python: list_data_quality_statistics)

Ruft eine Liste von Datenqualitätsstatistiken ab.

Anforderung

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.
- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Die Profil-ID.

- `TimestampFilter` – Ein TimestampFilter-Objekt.

  Ein Zeitstempelfilter.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der Ergebnisse, die bei dieser Anfrage zurückzugeben sind.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungstoken zum Anfordern der nächsten Ergebnisseite.

Antwort

- `Statistics` – Ein Array mit StatisticSummary-Objekten.

  Ein `StatisticSummaryList`

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungstoken zum Anfordern der nächsten Ergebnisseite.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

TimestampFilter Struktur

Ein Zeitstempelfilter.

Felder

- `RecordedBefore` – Zeitstempel.

  Der Zeitstempel, vor dem Statistiken in die Ergebnisse aufgenommen werden sollen.

- `RecordedAfter` – Zeitstempel.

Der Zeitstempel, nach dem Statistiken in die Ergebnisse aufgenommen werden sollen.

CreateDataQualityRulesetRequest Struktur

Eine Anfrage zur Erstellung eines Regelsatzes für die Datenqualität.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein eindeutiger Name für den Datenqualitätsregelsatz.
- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes für die Datenqualität.
- `Ruleset` – Erforderlich: UTF-8-Zeichenkette, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Eine Liste von Tags, die auf den Datenqualitätsregelsatz angewendet werden.
- `TargetTable` – Ein DataQualityTargetTable-Objekt.

  Eine Zieltabelle, die dem Datenqualitätsregelsatz zugeordnet ist.
- `RecommendationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ausführungs-ID für die Empfehlungsausführung.
- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Wird für Idempotenz verwendet und sollte auf eine zufällige ID (z. B. eine UUID) festgelegt werden, um zu vermeiden, dass mehrere Instances der gleichen Ressource erstellt oder gestartet werden.

GetDataQualityRulesetResponse Struktur

Gibt die Antwort auf den Regelsatz zur Datenqualität zurück.

Felder

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes.

- `Description` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Beschreibung des Regelsatzes.

- `Ruleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Ein Regelsatz der Definitionssprache für Datenqualität (DQDL). Weitere Informationen finden Sie im AWS Glue Entwicklerhandbuch.

- `TargetTable` – Ein DataQualityTargetTable-Objekt.

  Der Name und der Datenbankname der Zieltabelle.

- `CreatedOn` – Zeitstempel.

  Ein Zeitstempel. Die Uhrzeit und das Datum, an dem dieser Datenqualitätsregelsatz erstellt wurde.

- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der letzte Zeitpunkt, an dem dieser Datenqualitätsregelsatz geändert wurde.

- `RecommendationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Wenn ein Regelsatz aus einer Empfehlungsausführung erstellt wurde, wird diese Ausführungs-ID generiert, um die beiden miteinander zu verknüpfen.

- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

GetDataQualityResultResponse Struktur

Die Antwort auf das Datenqualitätsergebnis.

Felder

- `ResultId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine eindeutige Ergebnis-ID für das Datenqualitätsergebnis.

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID für das Datenqualitätsergebnis.

- `Score` – Zahl (Double), nicht mehr als 1,0.

  Ein aggregierter Datenqualitätswert. Stellt das Verhältnis der Regeln dar, die an die Gesamtzahl der Regeln übergeben wurden.

- `DataSource` – Ein DataSource-Objekt.

  Die Tabelle, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `RulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes, der dem Datenqualitätsergebnis zugeordnet ist.

- `EvaluationContext` – UTF-8-Zeichenfolge.

Im Kontext eines Jobs in AWS Glue Studio wird in der Regel jedem Knoten auf der Arbeitsfläche ein Name zugewiesen, und Datenqualitätsknoten haben Namen. Bei mehreren Knoten kann das `evaluationContext` die Knoten unterscheiden.

- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Ausführung für dieses Datenqualitätsergebnis gestartet wurde.

- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der die Ausführung für dieses Datenqualitätsergebnis abgeschlossen wurde.

- `JobName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Auftragsname, der dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `JobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Auftragsausführungs-ID, die dem Datenqualitätsergebnis zugeordnet ist, falls vorhanden.

- `RulesetEvaluationRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die eindeutige Ausführungs-ID, die der Regelsatzauswertung zugeordnet ist.

- `RuleResults` – Ein Array mit DataQualityRuleResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityRuleResult`-Objekten, die die Ergebnisse für jede Regel darstellen.

- `AnalyzerResults` – Ein Array mit DataQualityAnalyzerResult-Objekten, nicht mehr als 2000 Strukturen.

  Eine Liste von `DataQualityAnalyzerResult`-Objekten, die die Ergebnisse für jede Analysator darstellen.

- `Observations` – Ein Array mit DataQualityObservation-Objekten, nicht mehr als 50 Strukturen.

  Eine Liste von `DataQualityObservation`-Objekten, die die Beobachtungen darstellen, die nach der Auswertung der Regeln und Analysatoren generiert wurden.

- `AggregatedMetrics` – Ein DataQualityAggregatedMetrics-Objekt.

Eine Zusammenfassung der `DataQualityAggregatedMetrics` Objekte mit der Gesamtzahl der verarbeiteten Zeilen und Regeln, einschließlich ihrer pass/fail Statistiken, die auf Ergebnissen auf Zeilenebene basieren.

StartDataQualityRuleRecommendationRunRequest Struktur

Die Anfrage mit der Empfehlung zur Datenqualitätsregel.

Felder

- `DataSource` – Erforderlich: Ein DataSource-Objekt.

  Die diesem Lauf zugeordnete Datenquelle (AWS Glue Tabelle).

- `Role` – Erforderlich: UTF-8-Zeichenfolge.

  Eine IAM Rolle, die zur Verschlüsselung der Ergebnisse des Laufs bereitgestellt wird.

- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.

- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).

- `CreatedRulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Name für den Regelsatz.

- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Wird für Idempotenz verwendet und sollte auf eine zufällige ID (z. B. eine UUID) festgelegt werden, um zu vermeiden, dass mehrere Instances der gleichen Ressource erstellt oder gestartet werden.

GetDataQualityRuleRecommendationRunResponse Struktur

Die Antwort auf den Empfehlungslauf für die Datenqualitätsregel.

Felder

- `RunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die dieser Ausführung zugeordnete eindeutige Kennung.
- `DataSource` – Ein DataSource-Objekt.

  Die diesem Lauf zugeordnete Datenquelle (eine AWS Glue Tabelle).
- `Role` – UTF-8-Zeichenfolge.

  Eine IAM Rolle, die bereitgestellt wird, um die Ergebnisse des Laufs zu verschlüsseln.
- `NumberOfWorkers` – Zahl (Ganzzahl).

  Die Anzahl der `G.1X`-Worker, die in der Ausführung verwendet werden sollen. Der Standardwert ist 5.
- `Timeout` – Zahl (ganze Zahl), mindestens 1.

  Das Timeout für eine Ausführung in Minuten. Dies ist die maximale Zeitspanne, in der eine Ausführung Ressourcen verbrauchen kann, bevor diese beendet wird und in den `TIMEOUT`-Status wechselt. Der Standardwert beträgt 2 880 Minuten (48 Stunden).
- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Der Status für diese Ausführung.
- `ErrorString` – UTF-8-Zeichenfolge.

  Die Fehlerzeichenfolgen, die der Ausführung zugeordnet sind.
- `StartedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung gestartet wurde.

- `LastModifiedOn` – Zeitstempel.

  Ein Zeitstempel. Der letzte Zeitpunkt, an dem diese Empfehlungsausführung für die Datenqualitätsregel geändert wurde.
- `CompletedOn` – Zeitstempel.

  Das Datum und die Uhrzeit, zu der diese Ausführung abgeschlossen wurde.
- `ExecutionTime` – Zahl (Ganzzahl).

  Die Zeit (in Sekunden), in der durch die Ausführung Ressourcen verbraucht wurden.
- `RecommendedRuleset` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 65 536 Bytes lang.

  Wenn die Empfehlungsausführung einer Startregel abgeschlossen ist, wird ein empfohlener Regelsatz (ein Satz von Regeln) erstellt. Dieses Mitglied verfügt über diese Regeln im Format der Definitionssprache für Datenqualität (DQDL).
- `CreatedRulesetName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des Regelsatzes, der durch die Ausführung erstellt wurde.
- `DataQualitySecurityConfiguration` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Sicherheitskonfiguration, die mit der Datenqualitätsverschlüsselungsoption erstellt wurde.

BatchPutDataQualityStatisticAnnotation Aktion (Python: batch_put_data_quality_statistic_annotation)

Kommentieren Sie Datenpunkte im Zeitverlauf für eine bestimmte Datenqualitätsstatistik.

Anforderung

- `InclusionAnnotations` – Erforderlich: Ein Array mit DatapointInclusionAnnotation-Objekten.

  Eine Liste von. `DatapointInclusionAnnotation`
- `ClientToken` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

Client-Token.

Antwort

- `FailedInclusionAnnotations` – Ein Array mit AnnotationError-Objekten.

  Eine Liste `AnnotationError` von.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

GetDataQualityModel Aktion (Python: get_data_quality_model)

Rufen Sie den Trainingsstatus des Modells zusammen mit weiteren Informationen (,,) ab. CompletedOn StartedOn FailureReason

Anforderung

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.
- `ProfileId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID.

Antwort

- `Status` – UTF-8-Zeichenfolge (zulässige Werte: `RUNNING` | `SUCCEEDED` | `FAILED`).

  Der Trainingsstatus des Datenqualitätsmodells.
- `StartedOn` – Zeitstempel.

Der Zeitstempel, zu dem das Training des Datenqualitätsmodells begann.

- `CompletedOn` – Zeitstempel.

  Der Zeitstempel, zu dem das Training des Datenqualitätsmodells abgeschlossen wurde.

- `FailureReason` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Grund für das Fehlschlagen des Trainings.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityModelResult Aktion (Python: get_data_quality_model_result)

Ruft die Vorhersagen einer Statistik für eine bestimmte Profil-ID ab.

Anforderung

- `StatisticId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.

- `ProfileId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID.

Antwort

- `CompletedOn` – Zeitstempel.

  Der Zeitstempel, zu dem das Training des Datenqualitätsmodells abgeschlossen wurde.

- `Model` – Ein Array mit StatisticModelResult-Objekten.

  Eine Liste von `StatisticModelResult`

Fehler

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityStatisticAnnotations Aktion (Python: list_data_quality_statistic_annotations)

Rufen Sie Anmerkungen für eine Datenqualitätsstatistik ab.

Anforderung

- `StatisticId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Statistik-ID.

- `ProfileId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die Profil-ID.

- `TimestampFilter` – Ein TimestampFilter-Objekt.

  Ein Zeitstempelfilter.

- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der Ergebnisse, die bei dieser Anfrage zurückzugeben sind.

- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungstoken zum Abrufen der nächsten Ergebnisgruppe.

Antwort

- `Annotations` – Ein Array mit StatisticAnnotation-Objekten.

  Eine Liste der `StatisticAnnotation` auf die Statistik angewendeten
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungstoken zum Abrufen der nächsten Ergebnisgruppe.

Fehler

- `InvalidInputException`
- `InternalServiceException`

PutDataQualityProfileAnnotation Aktion (Python: put_data_quality_profile_annotation)

Kommentieren Sie alle Datenpunkte für ein Profil.

Anforderung

- `ProfileId` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID des Profils zur Überwachung der Datenqualität, das mit Anmerkungen versehen werden soll.
- `InclusionAnnotation` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `INCLUDE` | `EXCLUDE`).

  Der Wert der Inklusionsanmerkung, der auf das Profil angewendet werden soll.

Antwort

- Keine Antwortparameter.

Fehler

- `EntityNotFoundException`
- `InvalidInputException`

- `InternalServiceException`

API zur Erkennung sensibler Daten

Die API zur Erkennung sensibler Daten beschreibt, wie sensible Daten in den Spalten und Zeilen Ihrer strukturierten Daten erkannt werden. APIs

Datentypen

- CustomEntityType Struktur

CustomEntityType Struktur

Ein Objekt, das ein benutzerdefiniertes Muster zum Erkennen sensibler Daten in den Spalten und Zeilen Ihrer strukturierten Daten darstellt.

Felder

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Name für das benutzerdefinierte Muster, mit dem es später abgerufen oder gelöscht werden kann. Dieser Name muss pro AWS Konto eindeutig sein.
- `RegexString` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Zeichenfolge für reguläre Ausdrücke, die zum Erkennen sensibler Daten in einem benutzerdefinierten Muster verwendet wird.
- `ContextWords` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 20 Zeichenfolgen.

  Eine Liste von Kontextwörtern. Wenn keines dieser Kontextwörter in der Nähe des regulären Ausdrucks gefunden wird, werden die Daten nicht als sensible Daten erkannt.

  Wenn keine Kontextwörter übergeben werden, wird nur ein regulärer Ausdruck überprüft.

Operationen

- CreateCustomEntityType Aktion (Python: create_custom_entity_type)
- DeleteCustomEntityType Aktion (Python: delete_custom_entity_type)
- GetCustomEntityType Aktion (Python: get_custom_entity_type)
- BatchGetCustomEntityTypes Aktion (Python: batch_get_custom_entity_types)
- ListCustomEntityTypes Aktion (Python: list_custom_entity_types)

CreateCustomEntityType Aktion (Python: create_custom_entity_type)

Erstellt ein benutzerdefiniertes Muster, das verwendet wird, um sensible Daten in den Spalten und Zeilen Ihrer strukturierten Daten zu erkennen.

Jedes benutzerdefinierte Muster, das Sie erstellen, gibt einen regulären Ausdruck und eine optionale Liste von Kontextwörtern an. Wenn keine Kontextwörter übergeben werden, wird nur ein regulärer Ausdruck überprüft.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Ein Name für das benutzerdefinierte Muster, mit dem es später abgerufen oder gelöscht werden kann. Dieser Name muss pro Konto eindeutig sein. AWS
- `RegexString` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Zeichenfolge für reguläre Ausdrücke, die zum Erkennen sensibler Daten in einem benutzerdefinierten Muster verwendet wird.
- `ContextWords` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 20 Zeichenfolgen.

  Eine Liste von Kontextwörtern. Wenn keines dieser Kontextwörter in der Nähe des regulären Ausdrucks gefunden wird, werden die Daten nicht als sensible Daten erkannt.

  Wenn keine Kontextwörter übergeben werden, wird nur ein regulärer Ausdruck überprüft.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

Eine Liste von Tags, die auf den benutzerdefinierten Entitätstyp angewendet werden.

Antwort

- Name – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des benutzerdefinierten Musters, das Sie erstellt haben.

Fehler

- `AccessDeniedException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteCustomEntityType Aktion (Python: delete_custom_entity_type)

Löscht ein benutzerdefiniertes Muster durch Angabe seines Namens.

Anforderung

- Name – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des benutzerdefinierten Musters, das Sie löschen möchten.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des benutzerdefinierten Musters, das Sie gelöscht haben.

Fehler

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetCustomEntityType Aktion (Python: get_custom_entity_type)

Ruft die Details eines benutzerdefinierten Musters durch Angabe seines Namens ab.

Anforderung

- `Name` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des benutzerdefinierten Musters, das Sie abrufen möchten.

Antwort

- `Name` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name des benutzerdefinierten Musters, das Sie abgerufen haben.
- `RegexString` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Eine Zeichenfolge für reguläre Ausdrücke, die zum Erkennen sensibler Daten in einem benutzerdefinierten Muster verwendet wird.

- `ContextWords` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 20 Zeichenfolgen.

  Eine Liste von Kontextwörtern, falls angegeben, als Sie das benutzerdefinierte Muster erstellt haben. Wenn keines dieser Kontextwörter in der Nähe des regulären Ausdrucks gefunden wird, werden die Daten nicht als sensible Daten erkannt.

Fehler

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

BatchGetCustomEntityTypes Aktion (Python: batch_get_custom_entity_types)

Ruft die Details für die benutzerdefinierten Muster ab, die durch eine Namensliste angegeben sind.

Anforderung

- `Names` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 50 Zeichenfolgen.

  Eine Liste der Namen der benutzerdefinierten Muster, die Sie abrufen möchten.

Antwort

- `CustomEntityTypes` – Ein Array mit CustomEntityType-Objekten.

  Eine Liste der `CustomEntityType`-Objekte, die die erstellten benutzerdefinierten Muster darstellen.
- `CustomEntityTypesNotFound` – Ein Array mit UTF-8-Zeichenfolgen, nicht weniger als 1 und nicht mehr als 50 Zeichenfolgen.

  Eine Liste der Namen benutzerdefinierter Muster, die nicht gefunden werden konnten.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

ListCustomEntityTypes Aktion (Python: list_custom_entity_types)

Listet alle benutzerdefinierten Muster auf, die erstellt wurden.

Anforderung

- `NextToken` – UTF-8-Zeichenfolge.

  Ein paginiertes Token zum Ausgleich der Ergebnisse.
- `MaxResults` – Zahl (Ganzzahl), nicht kleiner als 1 oder größer als 1000.

  Die maximale Anzahl der auszugebenden Ergebnisse.
- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Eine Liste mit Tags für Schlüsselwertpaare.

Antwort

- `CustomEntityTypes` – Ein Array mit CustomEntityType-Objekten.

  Eine Liste der `CustomEntityType`-Objekte, die benutzerdefinierte Muster darstellen.
- `NextToken` – UTF-8-Zeichenfolge.

  Ein Paginierungs-Token, falls mehr Ergebnisse verfügbar sind.

Fehler

- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`

Einloggen APIs AWS Glue

Datentypen

- Tag-Struktur

Tag-Struktur

Das `Tag` Objekt stellt eine Bezeichnung dar, die Sie einer AWS Ressource zuweisen können. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen.

Weitere Informationen zu Tags und zur Steuerung des Zugriffs auf Ressourcen finden Sie unter AWS Tags in AWS Glue und Spezifizieren von AWS Glue Ressourcen ARNs im Entwicklerhandbuch. AWS Glue

Felder

- `key` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Tag-Schlüssel. Der Schlüssel ist erforderlich, wenn Sie ein Tag für ein Objekt erstellen. Bei dem Schlüssel wird die Groß-/Kleinschreibung berücksichtigt und er darf nicht das Präfix "aws" enthalten.
- `value` – UTF-8-Zeichenfolge, nicht mehr als 256 Bytes lang.

  Der Tag-Wert. Der Wert ist optional, wenn Sie ein Tag für ein Objekt erstellen. Bei dem Wert wird die Groß-/Kleinschreibung berücksichtigt und er darf nicht das Präfix "aws" enthalten.

Operationen

- TagResource Aktion (Python: tag_resource)
- UntagResource Aktion (Python: untag_resource)
- GetTags Aktion (Python: get_tags)

TagResource Aktion (Python: tag_resource)

Fügt einer Ressource Tags hinzu. Ein Tag ist eine Bezeichnung, die Sie einer AWS Ressource zuweisen können. In AWS Glue können Sie nur bestimmte Ressourcen taggen. Weitere Informationen dazu, welche Ressourcen Sie markieren können, finden Sie unter AWS -Tags in AWS Glue.

Zusätzlich zu den Tagging-Berechtigungen für das Aufrufen von Tag-Related benötigen Sie auch die `glue:GetConnection` Berechtigung APIs, Tagging für Verbindungen APIs aufzurufen, und die `glue:GetDatabase` Berechtigung, Tagging APIs für Datenbanken aufzurufen.

Anforderung

- `ResourceArn` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der ARN der AWS Glue Ressource, zu der die Tags hinzugefügt werden sollen. Weitere Informationen zur AWS Glue Ressource ARNs finden Sie im AWS Glue ARN-Zeichenkettenmuster.
- `TagsToAdd` – Erforderlich: Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

  Zu dieser Ressource hinzuzufügende Tags.

Antwort

- Keine Antwortparameter.

Fehler

- `ResourceNotFoundException`

UntagResource Aktion (Python: untag_resource)

Entfernt die angegebenen Tags aus einer Integrationsressource.

Anforderung

- `ResourceArn` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-Ressourcenname (ARN) für die Integrationsressource.

- `TagsToRemove` – Erforderlich: Ein Array mit UTF-8-Zeichenfolgen, nicht mehr als 50 Zeichenfolgen.

  Eine Liste von Metadaten-Tags, die aus der Ressource entfernt werden sollen.

Antwort

- Keine Antwortparameter.

Fehler

- `ResourceNotFoundException`

GetTags Aktion (Python: get_tags)

Ruft eine Liste von Tags ab, die mit einer Ressource verknüpft sind.

Anforderung

- `ResourceArn` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 10 240 Bytes lang, passend zum Custom string pattern #49.

  Der Amazon-ARN der Ressource, für die Tags abgerufen werden sollen.

Antwort

- `Tags` – Ein Map-Array von Schlüssel-Wert-Paaren, nicht mehr als 50 Paare.

  Jeder Schlüssel ist eine UTF-8-Zeichenfolge, die nicht weniger als 1 oder mehr als 128 Bytes lang ist.

  Jeder Wert ist eine UTF-8-Zeichenfolge, die nicht mehr als 256 Bytes lang ist.

Die angeforderten Tags.

Fehler

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

Gängige Datentypen

Die gängigen Datentypen beschreiben verschiedene Datentypen, die in AWS Glueüblich sind.

Tag-Struktur

Das `Tag` Objekt stellt eine Bezeichnung dar, die Sie einer AWS Ressource zuweisen können. Jeder Tag (Markierung) besteht aus einem Schlüssel und einem optionalen Wert, beides können Sie bestimmen.

Weitere Informationen zu Tags und zur Steuerung des Zugriffs auf Ressourcen finden Sie unter AWS Tags in AWS Glue und Spezifizieren von AWS Glue Ressourcen ARNs im Entwicklerhandbuch. AWS Glue

Felder

- `key` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 128 Bytes lang.

  Der Tag-Schlüssel. Der Schlüssel ist erforderlich, wenn Sie ein Tag für ein Objekt erstellen. Bei dem Schlüssel wird die Groß-/Kleinschreibung berücksichtigt und er darf nicht das Präfix "aws" enthalten.

- `value` – UTF-8-Zeichenfolge, nicht mehr als 256 Bytes lang.

  Der Tag-Wert. Der Wert ist optional, wenn Sie ein Tag für ein Objekt erstellen. Bei dem Wert wird die Groß-/Kleinschreibung berücksichtigt und er darf nicht das Präfix "aws" enthalten.

DecimalNumber Struktur

Enthält einen numerischen Wert im Dezimalformat.

Felder

- `UnscaledValue` – Erforderlich: Blob.

  Der nicht skalierte numerische Wert.
- `Scale` – Erforderlich: Zahl (Ganzzahl).

  Die Skalierung, die bestimmt, wo das Dezimalzeichen im nicht skalierten Wert gesetzt wird.

ErrorDetail Struktur

Enthält Details über einen Fehler.

Felder

- `ErrorCode` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Code im Zusammenhang mit diesem Fehler.
- `ErrorMessage` – Beschreibende Zeichenfolge, nicht mehr als 2048 Bytes lang, passend zum URI address multi-line string pattern.

  Eine Meldung mit einer Beschreibung des Fehlers.

PropertyPredicate Struktur

Definiert ein Eigenschaftsprädikat.

Felder

- `Key`— Wertezeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Der Schlüssel der Eigenschaft.
- `Value`— Wertezeichenfolge, nicht weniger als 1 oder mehr als 1024 Byte lang.

  Der Wert der Eigenschaft.

- `Comparator` – UTF-8-Zeichenfolge (zulässige Werte: `EQUALS` | `GREATER_THAN` | `LESS_THAN` | `GREATER_THAN_EQUALS` | `LESS_THAN_EQUALS`).

  Der Vergleichsoperator, mit dem diese Eigenschaft mit anderen verglichen wird.

ResourceUri Struktur

Die Ressourcen URIs für die Funktion.

Felder

- `ResourceType` – UTF-8-Zeichenfolge (zulässige Werte: `JAR` | `FILE` | `ARCHIVE`).

  Der Ressourcentyp.
- `Uri` – Uniform Resource Identifier (uri), nicht weniger als 1 oder mehr als 1024 Bytes lang, passend zum URI address multi-line string pattern.

  Der URI für den Zugriff auf die Ressource.

ColumnStatistics Struktur

Stellt die generierten Statistiken auf Spaltenebene für eine Tabelle oder Partition dar.

Felder

- `ColumnName` – Erforderlich: UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Name der Spalte, zu der Statistiken gehören.
- `ColumnType` – Erforderlich: Typ-Name, nicht mehr als 20 000 Bytes lang, passend zum Single-line string pattern.

  Der Datentyp der Spalte.
- `AnalyzedTime` – Erforderlich: Zeitstempel.

  Der Zeitstempel für die Generierung der Spaltenstatistiken.
- `StatisticsData` – Erforderlich: Ein ColumnStatisticsData-Objekt.

  Ein `ColumnStatisticData`-Objekt, das die Statistikdatenwerte enthält.

ColumnStatisticsError Struktur

Enthält ein fehlgeschlagenes `ColumnStatistics`-Objekt und den Grund für den Fehler.

Felder

- `ColumnStatistics` – Ein ColumnStatistics-Objekt.

  Die `ColumnStatistics` der Spalte.
- `Error` – Ein ErrorDetail-Objekt.

  Eine Fehlermeldung mit dem Grund für den Fehler eines Vorgangs.

ColumnError Struktur

Enthält den Namen der fehlgeschlagenen Spalte und den Grund für den Fehler.

Felder

- `ColumnName` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Der Name der Spalte, die fehlgeschlagen ist.
- `Error` – Ein ErrorDetail-Objekt.

  Eine Fehlermeldung mit dem Grund für den Fehler eines Vorgangs.

ColumnStatisticsData Struktur

Enthält die einzelnen Typen von Spaltenstatistikdaten. Es sollte nur ein Datenobjekt festgelegt und durch das `Type`-Attribut angegeben werden.

Felder

- `Type` – Erforderlich: UTF-8-Zeichenfolge (zulässige Werte: `BOOLEAN` | `DATE` | `DECIMAL` | `DOUBLE` | `LONG` | `STRING` | `BINARY`).

  Der Typ der Spaltenstatistikdaten.
- `BooleanColumnStatisticsData` – Ein BooleanColumnStatisticsData-Objekt.

Boolesche Spaltenstatistikdaten.

- `DateColumnStatisticsData` – Ein DateColumnStatisticsData-Objekt.

  Datum der Spaltenstatistikdaten.

- `DecimalColumnStatisticsData` – Ein DecimalColumnStatisticsData-Objekt.

  Statistikdaten in Dezimalspalten. UnscaledValues Darin befinden sich Base64-kodierte Binärobjekte, die Big-Endian-Darstellungen, also Zweierkomplementdarstellungen des unskalierten Dezimalwerts speichern.

- `DoubleColumnStatisticsData` – Ein DoubleColumnStatisticsData-Objekt.

  Double-Spaltenstatistikdaten.

- `LongColumnStatisticsData` – Ein LongColumnStatisticsData-Objekt.

  Lange Spaltenstatistikdaten.

- `StringColumnStatisticsData` – Ein StringColumnStatisticsData-Objekt.

  Zeichenfolgen-Spaltenstatistikdaten.

- `BinaryColumnStatisticsData` – Ein BinaryColumnStatisticsData-Objekt.

  Binäre Spaltenstatistikdaten.

BooleanColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für boolesche Datenspalten unterstützt werden.

Felder

- `NumberOfTrues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der wahren Werte in der Spalte.

- `NumberOfFalses` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der falschen Werte in der Spalte.

- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.

DateColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Zeitstempel-Datenspalten unterstützt werden.

Felder

- `MinimumValue` – Zeitstempel.

  Der niedrigste Wert in der Spalte.
- `MaximumValue` – Zeitstempel.

  Der höchste Wert in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.
- `NumberOfDistinctValues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der unterschiedlichen Werte in einer Spalte.

DecimalColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Festkommazahl-Datenspalten unterstützt werden.

Felder

- `MinimumValue` – Ein DecimalNumber-Objekt.

  Der niedrigste Wert in der Spalte.
- `MaximumValue` – Ein DecimalNumber-Objekt.

  Der höchste Wert in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.
- `NumberOfDistinctValues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der unterschiedlichen Werte in einer Spalte.

DoubleColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Gleitkommazahl-Datenspalten unterstützt werden.

Felder

- `MinimumValue` – Nummer (doppelt).

  Der niedrigste Wert in der Spalte.
- `MaximumValue` – Nummer (doppelt).

  Der höchste Wert in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.
- `NumberOfDistinctValues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der unterschiedlichen Werte in einer Spalte.

LongColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Ganzzahl-Datenspalten unterstützt werden.

Felder

- `MinimumValue` – Zahl (lang).

  Der niedrigste Wert in der Spalte.
- `MaximumValue` – Zahl (lang).

  Der höchste Wert in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.
- `NumberOfDistinctValues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der unterschiedlichen Werte in einer Spalte.

StringColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Zeichenfolge-Datenwerte unterstützt werden.

Felder

- `MaximumLength` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Größe der längsten Zeichenfolge in der Spalte.
- `AverageLength` – Erforderlich: Zahl (Double), nicht mehr als Keine.

  Die durchschnittliche Länge der Zeichenfolge in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.
- `NumberOfDistinctValues` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der unterschiedlichen Werte in einer Spalte.

BinaryColumnStatisticsData Struktur

Definiert Spaltenstatistiken, die für Bitfolge-Datenwerte unterstützt werden.

Felder

- `MaximumLength` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Größe der längsten Bitfolge in der Spalte.
- `AverageLength` – Erforderlich: Zahl (Double), nicht mehr als Keine.

  Die durchschnittliche Bitfolgelänge in der Spalte.
- `NumberOfNulls` – Erforderlich: Zahl (lang), nicht mehr als Keine.

  Die Anzahl der Nullwerte in der Spalte.

Zeichenfolgemuster

Die API verwendet die folgenden regulären Ausdrücke, um zu definieren, welche Inhalte für verschiedene Zeichenfolgenparameter und -mitglieder gültig sind:

- Einzeiliges Zeichenfolgenmuster – "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\t]*"
- Mehrzeilige Zeichenfolgenmuster für URI-Adressen – "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\n\t]*"
- Ein Logstash-Grok-Zeichenfolgenmuster – "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\t]*"
- Bezeichner-Zeichenfolgemuster – "[A-Za-z_][A-Za-z0-9_]*"
- AWS IAM-ARN-Zeichenfolgemuster – "arn:aws:iam::\d{12}:role/.*"
- Versionszeichenfolgemuster – "^[a-zA-Z0-9-_]+$"
- Protokollgruppen-Zeichenfolgemuster – "[\.\-_/#A-Za-z0-9]+"
- Protokollstream-Zeichenfolgemuster – "[^:*]*"
- Benutzerdefiniertes Zeichenfolgenmuster #10 – "[a-zA-Z0-9-_]+"
- Benutzerdefiniertes Zeichenfolgenmuster #11 – "[-a-zA-Z0-9+=/:_]*"
- Benutzerdefiniertes Zeichenfolgenmuster #12 – "[\S\s]*"
- Benutzerdefiniertes Zeichenfolgenmuster #13 – ".*\S.*"
- Benutzerdefiniertes Zeichenfolgenmuster #14 – "[a-zA-Z0-9-=._/@]+"
- Benutzerdefiniertes Zeichenfolgenmuster #15 – "[1-9][0-9]*|[1-9][0-9]*-[1-9][0-9]*"
- Benutzerdefiniertes Zeichenfolgenmuster #16 – "[A-Z][A-Za-z\.]+"
- Benutzerdefiniertes Zeichenfolgenmuster #17 – "[\S]*"
- Benutzerdefiniertes Zeichenfolgenmuster #18 – "[\w]*"
- Benutzerdefiniertes Zeichenfolgenmuster #19 – "arn:aws[a-z\-]*:iam::\d{12}:role/?[a-zA-Z_0-9+=,.@\-_/]+"
- Benutzerdefiniertes Zeichenfolgenmuster #20 – "subnet-[a-z0-9]+"
- Benutzerdefiniertes Zeichenfolgenmuster #21 – "\d{12}"
- Benutzerdefiniertes Zeichenfolgenmuster #22 – "([a-z]+)-([a-z]+-)?([a-z]+)-[0-9]+[a-z]+"
- Benutzerdefiniertes Zeichenfolgenmuster #23 – "[a-zA-Z0-9.-]*"
- Benutzerdefiniertes Zeichenfolgenmuster #24 – "arn:aws[a-z0-9\-]*:lambda:[a-z0-9\-]+:\d{12}:function:([\w\-]{1,64})"
- Benutzerdefiniertes Zeichenfolgenmuster #25 – "^(?!(.*[.\/\\]|aws:)).*$"
- Benutzerdefiniertes Zeichenfolgenmuster #26 – "[^\r\n]"

- Benutzerdefiniertes Zeichenfolgenmuster #27 – "^\w+\.\w+\.\w+$"
- Benutzerdefiniertes Zeichenfolgenmuster #28 – "^\w+\.\w+$"
- Benutzerdefiniertes Zeichenfolgenmuster #29 – "^$|arn:aws[a-z0-9-]*:kms:.*"
- Benutzerdefiniertes Zeichenfolgenmuster #30 – "arn:aws[^:]*:iam::[0-9]*:role/.+"
- Benutzerdefiniertes Zeichenfolgenmuster #31 – "[\.\-_A-Za-z0-9]+"
- Benutzerdefiniertes Zeichenfolgenmuster #32 – "^s3://([^/]+)/([^/]+/)*([^/]+)$"
- Benutzerdefiniertes Zeichenfolgenmuster #33 – ".*"
- Benutzerdefiniertes Zeichenfolgenmuster #34 – „^(Sun|Mon|Tue|Wed|Thu|Fri|Sat):([01]?[0-9]|2[0-3])$“
- Benutzerdefiniertes Zeichenfolgenmuster Nr. 35 – „[a-zA-Z0-9_.-]+“
- Benutzerdefiniertes Zeichenkettenmuster #36 — "^arn:aws(-(cn|us-gov|iso(-[bef])?))?:secretsmanager:.*$“
- Benutzerdefiniertes Zeichenkettenmuster #37 — "\S+“
- Benutzerdefiniertes Zeichenkettenmuster #38 — "^[\x20-\x7E]*$“
- Benutzerdefiniertes Zeichenkettenmuster #39 — "^([a-zA-Z0-9_=]+)\.([a-zA-Z0-9_=]+)\.([a-zA-Z0-9_\-\+\/=]*)“
- Benutzerdefiniertes Zeichenkettenmuster #40 — "^(https?)://[-a-zA-Z0-9+&@#/%?=~_|!:,.;]*[-a-zA-Z0-9+&@#/%=~_|]“
- Benutzerdefiniertes Zeichenkettenmuster #41 — "^(https?):\/\/[^\s/$.?#].[^\s]*$“
- Benutzerdefiniertes Zeichenkettenmuster #42 — "^subnet-[a-z0-9]+$“
- Benutzerdefiniertes Zeichenkettenmuster #43 — "[\p{L}\p{N}\p{P}]*“
- Benutzerdefiniertes Zeichenkettenmuster #44 — "[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}“
- Benutzerdefiniertes Zeichenkettenmuster #45 — "[a-zA-Z0-9-_$#.]+“
- Benutzerdefiniertes Zeichenkettenmuster #46 — "^\d{12}$“
- Benutzerdefiniertes Zeichenkettenmuster #47 — "^(\w+\.)+\w+$“
- Benutzerdefiniertes Zeichenkettenmuster #48 — "^([2-3]|3[.]9)$“
- Benutzerdefiniertes Zeichenkettenmuster #49 — "arn:aws(-(cn|us-gov|iso(-[bef])?))?:glue:.*“
- Benutzerdefiniertes Zeichenkettenmuster #50 — "(^arn:aws(-(cn|us-gov|iso(-[bef])?))?:iam::\w{12}:root)“

- Benutzerdefiniertes Zeichenkettenmuster #51 — `"^arn:aws(-(cn|us-gov|iso(-[bef])?))?:iam::[0-9]{12}:role/.+“`
- Benutzerdefiniertes Zeichenkettenmuster #52 — `"[\s\S]*“`
- Benutzerdefiniertes Zeichenkettenmuster #53 — `"([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF]|[^\S\r\n"'= ;])*“`
- Benutzerdefiniertes Zeichenkettenmuster #54 — `"^[A-Z\_]+$“`
- Benutzerdefiniertes Zeichenkettenmuster #55 — `"^[A-Za-z0-9]+$“`
- Benutzerdefiniertes Zeichenkettenmuster #56 — `"[*A-Za-z0-9_-]*“`
- Benutzerdefiniertes Zeichenkettenmuster #57 — `"([\u0020-\u007E\r\s\n])*“`
- Benutzerdefiniertes Zeichenkettenmuster #58 — `"[A-Za-z0-9_-]*“`
- Benutzerdefiniertes Zeichenkettenmuster #59 — `"([\u0009\u000B\u000C\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF])*“`
- Benutzerdefiniertes Zeichenkettenmuster #60 — `"([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\s])*“`
- Benutzerdefiniertes Zeichenkettenmuster #61 — `"([^\r\n])*“`

Ausnahmen

In diesem Abschnitt werden AWS Glue Ausnahmen beschrieben, anhand derer Sie die Ursache von Problemen finden und diese beheben können. Weitere Informationen zu HTTP-Fehlercodes und Zeichenfolgen für Ausnahmen in Zusammenhang mit Machine Learning finden Sie unter the section called “AWS Glue Ausnahmen für maschinelles Lernen”.

AccessDeniedException Struktur

Der Zugriff auf eine Ressource wurde verweigert.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

AlreadyExistsException Struktur

Eine Ressource, die erstellt oder hinzugefügt werden soll, ist bereits vorhanden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ConcurrentModificationException Struktur

Zwei Prozesse versuchen gleichzeitig, eine Ressource zu ändern.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ConcurrentRunsExceededException Struktur

Es werden zu viele Aufträge gleichzeitig ausgeführt.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

CrawlerNotRunningException Struktur

Der angegebene Crawler wird nicht ausgeführt.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

CrawlerRunningException Struktur

Die Operation kann nicht durchgeführt werden, da der Crawler bereits ausgeführt wird.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

CrawlerStoppingException Struktur

Der angegebene Crawler wird angehalten.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

EntityNotFoundException Struktur

Eine angegebene Entity ist nicht vorhanden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.
- `FromFederationSource` – Boolesch.

  Gibt an, ob sich die Ausnahme auf eine Verbundquelle bezieht.

FederationSourceException Struktur

Eine Verbundquelle ist fehlgeschlagen.

Felder

- `FederationSourceErrorCode`— UTF-8-String (gültige Werte: `AccessDeniedException` | | `EntityNotFoundException` | `InvalidCredentialsException` | `InvalidInputException` | `InvalidResponseException` | `OperationTimeoutException` | | `OperationNotSupportedException` `InternalServiceException` `PartialFailureException` |`ThrottlingException`).

Der Fehlercode des Problems.

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

FederationSourceRetryableException Struktur

Eine Verbundquelle ist fehlgeschlagen, der Vorgang kann jedoch wiederholt werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

GlueEncryptionException Struktur

Eine Verschlüsselungsoperation ist fehlgeschlagen.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IdempotentParameterMismatchException Struktur

Dieselbe eindeutige Kennung war zwei verschiedenen Datensätzen zugewiesen.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

IllegalWorkflowStateException Struktur

Der Workflow befindet sich in einem ungültigen Zustand, um eine angeforderte Operation auszuführen.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InternalServiceException Struktur

Es ist ein interner Servicefehler aufgetreten.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InvalidExecutionEngineException Struktur

Es wurde eine unbekannte oder ungültige Ausführungs-Engine angegeben.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InvalidInputException Struktur

Die bereitgestellte Eingabe war nicht gültig.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.
- `FromFederationSource` – Boolesch.

  Gibt an, ob sich die Ausnahme auf eine Verbundquelle bezieht.

InvalidStateException Struktur

Ein Fehler, der darauf hinweist, dass Ihre Daten einen ungültigen Zustand haben.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

InvalidTaskStatusTransitionException Struktur

Eine ordentlicher Übergang von einer Aufgabe zur nächsten ist fehlgeschlagen.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

JobDefinitionErrorException Struktur

Eine Auftragsdefinition ist nicht gültig.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

JobRunInTerminalStateException Struktur

Der Beendigungsstatus einer Auftragsausführung signalisiert einen Fehler.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

JobRunInvalidStateTransitionException Struktur

Bei einer Auftragsausführung ist ein ungültiger Übergang vom Ausgangsstatus zum Zielstatus aufgetreten.

Felder

- `jobRunId` – UTF-8-Zeichenfolge, nicht weniger als 1 oder mehr als 255 Bytes lang, passend zum Single-line string pattern.

  Die ID der jeweiligen Auftragsausführung.
- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.
- `sourceState`— UTF-8-String (gültige Werte: `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  Der Ausgangsstatus.
- `targetState`— UTF-8-Zeichenfolge (gültige Werte: `STARTING` | | | | `RUNNING` | | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` `TIMEOUT` |`ERROR`). `WAITING` `EXPIRED`

  Der Zielstatus.

JobRunNotInTerminalStateException Struktur

Eine Auftragsausführung befindet sich nicht im Beendigungsstatus.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

LateRunnerException Struktur

Eine Auftragsausführung ist verspätet.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

NoScheduleException Struktur

Es gibt keinen entsprechenden Zeitplan.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

OperationTimeoutException Struktur

Bei der Operation ist eine Zeitüberschreitung aufgetreten.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ResourceNotReadyException Struktur

Eine Ressource war für eine Transaktion nicht bereit.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

ResourceNumberLimitExceededException Struktur

Der numerische Grenzwert einer Ressource wurde überschritten.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

SchedulerNotRunningException Struktur

Der angegebene Scheduler wird nicht ausgeführt.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

SchedulerRunningException Struktur

Der angegebene Scheduler wird bereits ausgeführt.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

SchedulerTransitioningException Struktur

Der angegebene Scheduler befindet sich im Übergang.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

UnrecognizedRunnerException Struktur

Die Auftragsausführung wurde nicht erkannt.

Felder

- `Message` – UTF-8-Zeichenfolge.

Eine Meldung mit einer Beschreibung des Problems.

ValidationException Struktur

Ein Wert konnte nicht überprüft werden.

Felder

- `Message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

VersionMismatchException Struktur

Es liegt ein Versionskonflikt vor.

Felder

- `message` – UTF-8-Zeichenfolge.

  Eine Meldung mit einer Beschreibung des Problems.

AWS Glue API-Codebeispiele mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Verwendung AWS Glue mit einem AWS Software Development Kit (SDK) funktioniert.

Bei Grundlagen handelt es sich um Code-Beispiele, die Ihnen zeigen, wie Sie die wesentlichen Vorgänge innerhalb eines Services ausführen.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarios anzeigen.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte

Hallo AWS Glue

Die folgenden Codebeispiele veranschaulichen, wie Sie mit der Verwendung von AWS Glue beginnen.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
namespace GlueActions;

public class HelloGlue
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
```

```
        // Set up dependency injection for AWS Glue.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonGlue>()
                .AddTransient<GlueWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloGlue>();
        var glueClient = host.Services.GetRequiredService<IAmazonGlue>();

        var request = new ListJobsRequest();

        var jobNames = new List<string>();

        do
        {
            var response = await glueClient.ListJobsAsync(request);
            jobNames.AddRange(response.JobNames);
            request.NextToken = response.NextToken;
        }
        while (request.NextToken is not null);

        Console.Clear();
        Console.WriteLine("Hello, Glue. Let's list your existing Glue Jobs:");
        if (jobNames.Count == 0)
        {
            Console.WriteLine("You don't have any AWS Glue jobs.");
        }
        else
        {
            jobNames.ForEach(Console.WriteLine);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Code für die CMake Datei CMake Lists.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS glue)

# Set this project's name.
project("hello_glue")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})
```

```
if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
                                     # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_glue.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code für die Quelldatei hello_glue.cpp.

```
#include <aws/core/Aws.h>
#include <aws/glue/GlueClient.h>
#include <aws/glue/model/ListJobsRequest.h>
#include <iostream>

/*
 *  A "Hello Glue" starter application which initializes an AWS Glue client and
 lists the
 *  AWS Glue job definitions.
 *
 *  main function
 *
 *  Usage: 'hello_glue'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
```

```
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::Glue::GlueClient glueClient(clientConfig);

        std::vector<Aws::String> jobs;

        Aws::String nextToken;  // Used for pagination.
        do {
            Aws::Glue::Model::ListJobsRequest listJobsRequest;
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }

            Aws::Glue::Model::ListJobsOutcome listRunsOutcome =
glueClient.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                jobs.insert(jobs.end(), jobNames.begin(), jobNames.end());

                nextToken = listRunsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
                result = 1;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "Your account has " << jobs.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < jobs.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << jobs[i] << std::endl;
        }
    }
    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
```

```
}
```

- Einzelheiten zur API finden Sie ListJobsunter AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
package com.example.glue;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.ListJobsRequest;
import software.amazon.awssdk.services.glue.model.ListJobsResponse;
import java.util.List;

public class HelloGlue {
    public static void main(String[] args) {
        GlueClient glueClient = GlueClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listJobs(glueClient);
    }

    public static void listJobs(GlueClient glueClient) {
        ListJobsRequest request = ListJobsRequest.builder()
                .maxResults(10)
                .build();
        ListJobsResponse response = glueClient.listJobs(request);
        List<String> jobList = response.jobNames();
        jobList.forEach(job -> {
            System.out.println("Job Name: " + job);
        });
    }
```

```
}
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import { ListJobsCommand, GlueClient } from "@aws-sdk/client-glue";

const client = new GlueClient({});

export const main = async () => {
  const command = new ListJobsCommand({});

  const { JobNames } = await client.send(command);
  const formattedJobNames = JobNames.join("\n");
  console.log("Job names: ");
  console.log(formattedJobNames);
  return JobNames;
};
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import boto3
from botocore.exceptions import ClientError


def hello_glue():
    """
    Lists the job definitions in your AWS Glue account, using the AWS SDK for
 Python (Boto3).
    """
    try:
        # Create the Glue client
        glue = boto3.client("glue")

        # List the jobs, limiting the results to 10 per page
        paginator = glue.get_paginator("get_jobs")
        response_iterator = paginator.paginate(
            PaginationConfig={"MaxItems": 10, "PageSize": 10}
        )

        # Print the job names
        print("Here are the jobs in your account:")
        for page in response_iterator:
            for job in page["Jobs"]:
                print(f"\t{job['Name']}")

    except ClientError as e:
        print(f"Error: {e}")


if __name__ == "__main__":
    hello_glue()
```

- Einzelheiten zur API finden Sie ListJobsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
require 'aws-sdk-glue'
require 'logger'

# GlueManager is a class responsible for managing AWS Glue operations
# such as listing all Glue jobs in the current AWS account.
class GlueManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all Glue jobs in the current AWS account.
  def list_jobs
    @logger.info('Here are the Glue jobs in your account:')

    paginator = @client.get_jobs(max_results: 10)
    jobs = []

    paginator.each_page do |page|
      jobs.concat(page.jobs)
    end

    if jobs.empty?
      @logger.info("You don't have any Glue jobs.")
    else
      jobs.each do |job|
```

```
        @logger.info("- #{job.name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  glue_client = Aws::Glue::Client.new
  manager = GlueManager.new(glue_client)
  manager.list_jobs
end
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Einzelheiten zur API finden Sie ListJobsin der API-Referenz zum AWS SDK für Rust.

AWS Glue API-Codebeispiele

- Grundlegende Beispiele für die AWS Glue Verwendung AWS SDKs
  - Hallo AWS Glue
  - Lernen Sie die Grundlagen AWS Glue mit einem AWS SDK kennen
  - Aktionen zur AWS Glue Verwendung AWS SDKs
    - Verwendung CreateCrawler mit einem AWS SDK
    - Verwendung CreateJob mit einem AWS SDK oder CLI
    - Verwendung DeleteCrawler mit einem AWS SDK
    - Verwendung DeleteDatabase mit einem AWS SDK
    - Verwendung DeleteJob mit einem AWS SDK oder CLI
    - Verwendung DeleteTable mit einem AWS SDK
    - Verwendung GetCrawler mit einem AWS SDK
    - Verwendung GetDatabase mit einem AWS SDK
    - Verwendung GetDatabases mit einem AWS SDK oder CLI
    - Verwendung GetJob mit einem AWS SDK oder CLI
    - Verwendung GetJobRun mit einem AWS SDK oder CLI
    - Verwendung GetJobRuns mit einem AWS SDK oder CLI
    - Verwendung GetTables mit einem AWS SDK oder CLI
    - Verwendung ListJobs mit einem AWS SDK
    - Verwendung StartCrawler mit einem AWS SDK oder CLI
    - Verwendung StartJobRun mit einem AWS SDK oder CLI

Grundlegende Beispiele für die AWS Glue Verwendung AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Grundlagen von AWS Glue with verwendet AWS SDKs werden.

Beispiele

- Hallo AWS Glue
- Lernen Sie die Grundlagen AWS Glue mit einem AWS SDK kennen
- Aktionen zur AWS Glue Verwendung AWS SDKs
  - Verwendung CreateCrawler mit einem AWS SDK

- Verwendung CreateJob mit einem AWS SDK oder CLI
- Verwendung DeleteCrawler mit einem AWS SDK
- Verwendung DeleteDatabase mit einem AWS SDK
- Verwendung DeleteJob mit einem AWS SDK oder CLI
- Verwendung DeleteTable mit einem AWS SDK
- Verwendung GetCrawler mit einem AWS SDK
- Verwendung GetDatabase mit einem AWS SDK
- Verwendung GetDatabases mit einem AWS SDK oder CLI
- Verwendung GetJob mit einem AWS SDK oder CLI
- Verwendung GetJobRun mit einem AWS SDK oder CLI
- Verwendung GetJobRuns mit einem AWS SDK oder CLI
- Verwendung GetTables mit einem AWS SDK oder CLI
- Verwendung ListJobs mit einem AWS SDK
- Verwendung StartCrawler mit einem AWS SDK oder CLI
- Verwendung StartJobRun mit einem AWS SDK oder CLI

Hallo AWS Glue

Die folgenden Codebeispiele veranschaulichen, wie Sie mit der Verwendung von AWS Glue beginnen.

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
namespace GlueActions;

public class HelloGlue
{
```

```
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for AWS Glue.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonGlue>()
                .AddTransient<GlueWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloGlue>();
        var glueClient = host.Services.GetRequiredService<IAmazonGlue>();

        var request = new ListJobsRequest();

        var jobNames = new List<string>();

        do
        {
            var response = await glueClient.ListJobsAsync(request);
            jobNames.AddRange(response.JobNames);
            request.NextToken = response.NextToken;
        }
        while (request.NextToken is not null);

        Console.Clear();
        Console.WriteLine("Hello, Glue. Let's list your existing Glue Jobs:");
        if (jobNames.Count == 0)
        {
            Console.WriteLine("You don't have any AWS Glue jobs.");
        }
        else
        {
            jobNames.ForEach(Console.WriteLine);
        }
```

```
    }
}
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Code für die CMake Datei CMake Lists.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS glue)

# Set this project's name.
project("hello_glue")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()
```

```
# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
                                     # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_glue.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code für die Quelldatei hello_glue.cpp.

```
#include <aws/core/Aws.h>
#include <aws/glue/GlueClient.h>
#include <aws/glue/model/ListJobsRequest.h>
#include <iostream>

/*
 *  A "Hello Glue" starter application which initializes an AWS Glue client and
 lists the
 *  AWS Glue job definitions.
 *
 *  main function
 *
 *  Usage: 'hello_glue'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
```

```
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::Glue::GlueClient glueClient(clientConfig);

        std::vector<Aws::String> jobs;

        Aws::String nextToken;  // Used for pagination.
        do {
            Aws::Glue::Model::ListJobsRequest listJobsRequest;
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }

            Aws::Glue::Model::ListJobsOutcome listRunsOutcome =
 glueClient.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
 listRunsOutcome.GetResult().GetJobNames();
                jobs.insert(jobs.end(), jobNames.begin(), jobNames.end());

                nextToken = listRunsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
                result = 1;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "Your account has " << jobs.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < jobs.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << jobs[i] << std::endl;
        }
```

```
    }
    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Einzelheiten zur API finden Sie ListJobsunter AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
package com.example.glue;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.ListJobsRequest;
import software.amazon.awssdk.services.glue.model.ListJobsResponse;
import java.util.List;

public class HelloGlue {
    public static void main(String[] args) {
        GlueClient glueClient = GlueClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listJobs(glueClient);
    }

    public static void listJobs(GlueClient glueClient) {
        ListJobsRequest request = ListJobsRequest.builder()
                .maxResults(10)
                .build();
        ListJobsResponse response = glueClient.listJobs(request);
        List<String> jobList = response.jobNames();
        jobList.forEach(job -> {
```

```
            System.out.println("Job Name: " + job);
        });
    }
}
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import { ListJobsCommand, GlueClient } from "@aws-sdk/client-glue";

const client = new GlueClient({});

export const main = async () => {
  const command = new ListJobsCommand({});

  const { JobNames } = await client.send(command);
  const formattedJobNames = JobNames.join("\n");
  console.log("Job names: ");
  console.log(formattedJobNames);
  return JobNames;
};
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für JavaScript API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import boto3
from botocore.exceptions import ClientError


def hello_glue():
    """
    Lists the job definitions in your AWS Glue account, using the AWS SDK for
 Python (Boto3).
    """
    try:
        # Create the Glue client
        glue = boto3.client("glue")

        # List the jobs, limiting the results to 10 per page
        paginator = glue.get_paginator("get_jobs")
        response_iterator = paginator.paginate(
            PaginationConfig={"MaxItems": 10, "PageSize": 10}
        )

        # Print the job names
        print("Here are the jobs in your account:")
        for page in response_iterator:
            for job in page["Jobs"]:
                print(f"\t{job['Name']}")

    except ClientError as e:
        print(f"Error: {e}")


if __name__ == "__main__":
    hello_glue()
```

- Einzelheiten zur API finden Sie ListJobsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
require 'aws-sdk-glue'
require 'logger'

# GlueManager is a class responsible for managing AWS Glue operations
# such as listing all Glue jobs in the current AWS account.
class GlueManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all Glue jobs in the current AWS account.
  def list_jobs
    @logger.info('Here are the Glue jobs in your account:')

    paginator = @client.get_jobs(max_results: 10)
    jobs = []

    paginator.each_page do |page|
      jobs.concat(page.jobs)
    end

    if jobs.empty?
      @logger.info("You don't have any Glue jobs.")
    else
      jobs.each do |job|
```

```
        @logger.info("- #{job.name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  glue_client = Aws::Glue::Client.new
  manager = GlueManager.new(glue_client)
  manager.list_jobs
end
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Einzelheiten zur API finden Sie ListJobsin der API-Referenz zum AWS SDK für Rust.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Lernen Sie die Grundlagen AWS Glue mit einem AWS SDK kennen

Die folgenden Code-Beispiele veranschaulichen Folgendes:

- Erstellen Sie einen Crawler, der einen öffentlichen Amazon-S3-Bucket crawlt und eine Datenbank mit CSV-formatierten Metadaten generiert.
- Listet Informationen zu Datenbanken und Tabellen in Ihrem auf AWS Glue Data Catalog.
- Erstellen Sie einen Auftrag, um CSV-Daten aus dem S3-Bucket zu extrahieren, die Daten umzuwandeln und die JSON-formatierte Ausgabe in einen anderen S3-Bucket zu laden.
- Listen Sie Informationen zu Auftragsausführungen auf, zeigen Sie transformierte Daten an und bereinigen Sie Ressourcen.

Weitere Informationen finden Sie unter Tutorial: Erste Schritte mit AWS Glue Studio.

.NET

SDK für .NET

> **Note**
>
> Es gibt mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie eine Klasse, die AWS Glue Funktionen umschließt, die im Szenario verwendet werden.

```
using System.Net;

namespace GlueActions;

public class GlueWrapper
{
    private readonly IAmazonGlue _amazonGlue;
```

```
    /// <summary>
    /// Constructor for the AWS Glue actions wrapper.
    /// </summary>
    /// <param name="amazonGlue"></param>
    public GlueWrapper(IAmazonGlue amazonGlue)
    {
        _amazonGlue = amazonGlue;
    }

    /// <summary>
    /// Create an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name for the crawler.</param>
    /// <param name="crawlerDescription">A description of the crawler.</param>
    /// <param name="role">The AWS Identity and Access Management (IAM) role to
    /// be assumed by the crawler.</param>
    /// <param name="schedule">The schedule on which the crawler will be
executed.</param>
    /// <param name="s3Path">The path to the Amazon Simple Storage Service
(Amazon S3)
    /// bucket where the Python script has been stored.</param>
    /// <param name="dbName">The name to use for the database that will be
    /// created by the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateCrawlerAsync(
        string crawlerName,
        string crawlerDescription,
        string role,
        string schedule,
        string s3Path,
        string dbName)
    {
        var s3Target = new S3Target
        {
            Path = s3Path,
        };

        var targetList = new List<S3Target>
        {
            s3Target,
        };

        var targets = new CrawlerTargets
```

```
        {
            S3Targets = targetList,
        };

        var crawlerRequest = new CreateCrawlerRequest
        {
            DatabaseName = dbName,
            Name = crawlerName,
            Description = crawlerDescription,
            Targets = targets,
            Role = role,
            Schedule = schedule,
        };

        var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="roleName">The name of the IAM role to be assumed by
    /// the job.</param>
    /// <param name="description">A description of the job.</param>
    /// <param name="scriptUrl">The URL to the script.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
    {
        var command = new JobCommand
        {
            PythonVersion = "3",
            Name = "glueetl",
            ScriptLocation = scriptUrl,
        };

        var arguments = new Dictionary<string, string>
        {
            { "--input_database", dbName },
            { "--input_table", tableName },
            { "--output_bucket_url", bucketUrl }
```

```
        };

        var request = new CreateJobRequest
        {
            Command = command,
            DefaultArguments = arguments,
            Description = description,
            GlueVersion = "3.0",
            Name = jobName,
            NumberOfWorkers = 10,
            Role = roleName,
            WorkerType = "G.1X"
        };

        var response = await _amazonGlue.CreateJobAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteCrawlerAsync(string crawlerName)
    {
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete the AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Get information about an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Crawler object describing the crawler.</returns>
    public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
```

```
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
 found.");
        return null;
    }


    /// <summary>
    /// Get information about the state of an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A value describing the state of the crawler.</returns>
    public async Task<CrawlerState> GetCrawlerStateAsync(string crawlerName)
    {
        var response = await _amazonGlue.GetCrawlerAsync(
            new GetCrawlerRequest { Name = crawlerName });
        return response.Crawler.State;
    }


    /// <summary>
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }


    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
```

```
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
{ JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }


    /// <summary>
    /// Get information about all AWS Glue runs of a specific job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A list of JobRun objects.</returns>
    public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
    {
        var jobRuns = new List<JobRun>();

        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };

        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }


    /// <summary>
    /// Get a list of tables for an AWS Glue database.
    /// </summary>
```

```
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A list of Table objects.</returns>
    public async Task<List<Table>> GetTablesAsync(string dbName)
    {
        var request = new GetTablesRequest { DatabaseName = dbName };
        var tables = new List<Table>();

        // Get a paginator for listing the tables.
        var tablePaginator = _amazonGlue.Paginators.GetTables(request);

        await foreach (var response in tablePaginator.Responses)
        {
            tables.AddRange(response.TableList);
        }

        return tables;
    }


    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }


    /// <summary>
    /// Start an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> StartCrawlerAsync(string crawlerName)
```

```
    {
        var crawlerRequest = new StartCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Start an AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A string representing the job run Id.</returns>
    public async Task<string> StartJobRunAsync(
        string jobName,
        string inputDatabase,
        string inputTable,
        string bucketName)
    {
        var request = new StartJobRunRequest
        {
            JobName = jobName,
            Arguments = new Dictionary<string, string>
            {
                {"--input_database", inputDatabase},
                {"--input_table", inputTable},
                {"--output_bucket_url", $"s3://{bucketName}/"}
            }
        };

        var response = await _amazonGlue.StartJobRunAsync(request);
        return response.JobRunId;
    }

}
```

Erstellen Sie eine Klasse, die das Szenario ausführt.

```
global using Amazon.Glue;
global using GlueActions;
global using Microsoft.Extensions.Configuration;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;



using Amazon.Glue.Model;
using Amazon.S3;
using Amazon.S3.Model;

namespace GlueBasics;

public class GlueBasics
{
    private static ILogger logger = null!;
    private static IConfiguration _configuration = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for AWS Glue.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonGlue>()
            .AddTransient<GlueWrapper>()
            .AddTransient<UiWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
        .CreateLogger<GlueBasics>();

        _configuration = new ConfigurationBuilder()
```

```
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // These values are stored in settings.json
        // Once you have run the CDK script to deploy the resources,
        // edit the file to set "BucketName", "RoleName", and "ScriptURL"
        // to the appropriate values. Also set "CrawlerName" to the name
        // you want to give the crawler when it is created.
        string bucketName = _configuration["BucketName"]!;
        string bucketUrl = _configuration["BucketUrl"]!;
        string crawlerName = _configuration["CrawlerName"]!;
        string roleName = _configuration["RoleName"]!;
        string sourceData = _configuration["SourceData"]!;
        string dbName = _configuration["DbName"]!;
        string cron = _configuration["Cron"]!;
        string scriptUrl = _configuration["ScriptURL"]!;
        string jobName = _configuration["JobName"]!;

        var wrapper = host.Services.GetRequiredService<GlueWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UiWrapper>();

        uiWrapper.DisplayOverview();
        uiWrapper.PressEnter();

        // Create the crawler and wait for it to be ready.
        uiWrapper.DisplayTitle("Create AWS Glue crawler");
        Console.WriteLine("Let's begin by creating the AWS Glue crawler.");

        var crawlerDescription = "Crawler created for the AWS Glue Basics
scenario.";
        var crawlerCreated = await wrapper.CreateCrawlerAsync(crawlerName,
crawlerDescription, roleName, cron, sourceData, dbName);
        if (crawlerCreated)
        {
            Console.WriteLine($"The crawler: {crawlerName} has been created. Now
let's wait until it's ready.");
            CrawlerState crawlerState;
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
```

```
            while (crawlerState != "READY");
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't create crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.DisplayTitle("Start AWS Glue crawler");
        Console.WriteLine("Now let's wait until the crawler has successfully
started.");
        var crawlerStarted = await wrapper.StartCrawlerAsync(crawlerName);
        if (crawlerStarted)
        {
            CrawlerState crawlerState;
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
            while (crawlerState != "READY");
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't start the crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.PressEnter();

        Console.WriteLine($"\nLet's take a look at the database: {dbName}");
        var database = await wrapper.GetDatabaseAsync(dbName);

        if (database != null)
        {
            uiWrapper.DisplayTitle($"{database.Name} Details");
            Console.WriteLine($"{database.Name} created on
{database.CreateTime}");
            Console.WriteLine(database.Description);
        }
```

```
        uiWrapper.PressEnter();

        var tables = await wrapper.GetTablesAsync(dbName);
        if (tables.Count > 0)
        {
            tables.ForEach(table =>
            {
                Console.WriteLine($"{table.Name}\tCreated:
{table.CreateTime}\tUpdated: {table.UpdateTime}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create AWS Glue job");
        Console.WriteLine("Creating a new AWS Glue job.");
        var description = "An AWS Glue job created using the AWS SDK for .NET";
        await wrapper.CreateJobAsync(dbName, tables[0].Name, bucketUrl, jobName,
roleName, description, scriptUrl);

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Starting AWS Glue job");
        Console.WriteLine("Starting the new AWS Glue job...");
        var jobRunId = await wrapper.StartJobRunAsync(jobName, dbName,
tables[0].Name, bucketName);
        var jobRunComplete = false;
        var jobRun = new JobRun();
        do
        {
            jobRun = await wrapper.GetJobRunAsync(jobName, jobRunId);
            if (jobRun.JobRunState == "SUCCEEDED" || jobRun.JobRunState ==
"STOPPED" ||
                jobRun.JobRunState == "FAILED" || jobRun.JobRunState ==
"TIMEOUT")
            {
                jobRunComplete = true;
            }
        } while (!jobRunComplete);

        uiWrapper.DisplayTitle($"Data in {bucketName}");

        // Get the list of data stored in the S3 bucket.
        var s3Client = new AmazonS3Client();
```

```
        var response = await s3Client.ListObjectsAsync(new ListObjectsRequest
{ BucketName = bucketName });
        response.S3Objects.ForEach(s3Object =>
        {
            Console.WriteLine(s3Object.Key);
        });

        uiWrapper.DisplayTitle("AWS Glue jobs");
        var jobNames = await wrapper.ListJobsAsync();
        jobNames.ForEach(jobName =>
        {
            Console.WriteLine(jobName);
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Get AWS Glue job run information");
        Console.WriteLine("Getting information about the AWS Glue job.");
        var jobRuns = await wrapper.GetJobRunsAsync(jobName);

        jobRuns.ForEach(jobRun =>
        {

Console.WriteLine($"{jobRun.JobName}\t{jobRun.JobRunState}\t{jobRun.CompletedOn}");
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Deleting resources");
        Console.WriteLine("Deleting the AWS Glue job used by the example.");
        await wrapper.DeleteJobAsync(jobName);

        Console.WriteLine("Deleting the tables from the database.");
        tables.ForEach(async table =>
        {
            await wrapper.DeleteTableAsync(dbName, table.Name);
        });

        Console.WriteLine("Deleting the database.");
        await wrapper.DeleteDatabaseAsync(dbName);

        Console.WriteLine("Deleting the AWS Glue crawler.");
        await wrapper.DeleteCrawlerAsync(crawlerName);
```

```
        Console.WriteLine("The AWS Glue scenario has completed.");
        uiWrapper.PressEnter();
    }
}


namespace GlueBasics;

public class UiWrapper
{
    public readonly string SepBar = new string('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the scenario.
    /// </summary>
    public void DisplayOverview()
    {
        Console.Clear();
        DisplayTitle("Amazon Glue: get started with crawlers and jobs");

        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t 1. Create a crawler, pass it the IAM role and the
 URL to the public S3 bucket that contains the source data");
        Console.WriteLine("\t 2. Start the crawler.");
        Console.WriteLine("\t 3. Get the database created by the crawler and the
 tables in the database.");
        Console.WriteLine("\t 4. Create a job.");
        Console.WriteLine("\t 5. Start a job run.");
        Console.WriteLine("\t 6. Wait for the job run to complete.");
        Console.WriteLine("\t 7. Show the data stored in the bucket.");
        Console.WriteLine("\t 8. List jobs for the account.");
        Console.WriteLine("\t 9. Get job run details for the job that was run.");
        Console.WriteLine("\t10. Delete the demo job.");
        Console.WriteLine("\t11. Delete the database and tables created for the
 demo.");
        Console.WriteLine("\t12. Delete the crawler.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
```

```
        Console.Write("\nPlease press <Enter> to continue. ");
        _ = Console.ReadLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to center on the screen.</param>
    /// <returns>The string padded to make it center on the screen.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für .NET -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase

- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
//! Scenario which demonstrates using AWS Glue to add a crawler and run a job.
/*!
 \\sa runGettingStartedWithGlueScenario()
 \param bucketName: An S3 bucket created in the setup.
 \param roleName: An AWS Identity and Access Management (IAM) role created in the
 setup.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */

bool AwsDoc::Glue::runGettingStartedWithGlueScenario(const Aws::String
 &bucketName,
                                                     const Aws::String &roleName,
                                                     const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::Glue::GlueClient client(clientConfig);

    Aws::String roleArn;
    if (!getRoleArn(roleName, roleArn, clientConfig)) {
```

```
        std::cerr << "Error getting role ARN for role." << std::endl;
        return false;
    }

    // 1. Upload the job script to the S3 bucket.
    {
        std::cout << "Uploading the job script '"
                  << AwsDoc::Glue::PYTHON_SCRIPT
                  << "'." << std::endl;

        if (!AwsDoc::Glue::uploadFile(bucketName,
                                      AwsDoc::Glue::PYTHON_SCRIPT_PATH,
                                      AwsDoc::Glue::PYTHON_SCRIPT,
                                      clientConfig)) {
            std::cerr << "Error uploading the job file." << std::endl;
            return false;
        }
    }

    // 2. Create a crawler.
    {
        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
```

```
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
        }
    }

    // 3. Get a crawler.
    {
        Aws::Glue::Model::GetCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

        if (outcome.IsSuccess()) {
            Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
            std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                              crawlerState)
                      << "." << std::endl;
        }
        else {
            std::cerr << "Error retrieving a crawler.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 4. Start a crawler.
    {
        Aws::Glue::Model::StartCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


        if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
                                    outcome.GetError().GetErrorType())) {
            if (!outcome.IsSuccess()) {
                std::cout << "Crawler was already started." << std::endl;
```

```
            }
            else {
                std::cout << "Successfully started crawler." << std::endl;
            }

            std::cout << "This may take a while to run." << std::endl;

            Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
            int iterations = 0;
            while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
                std::this_thread::sleep_for(std::chrono::seconds(1));
                ++iterations;
                if ((iterations % 10) == 0) { // Log status every 10 seconds.
                    std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                        crawlerState)
                              << ". After " << iterations
                              << " seconds elapsed."
                              << std::endl;
                }
                Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
                getCrawlerRequest.SetName(CRAWLER_NAME);

                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }

            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
```

```
                        << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 5. Get a database.
    {
        Aws::Glue::Model::GetDatabaseRequest request;
        request.SetName(CRAWLER_DATABASE_NAME);

        Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

        if (outcome.IsSuccess()) {
            const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

            std::cout << "Successfully retrieve the database\n" <<
                      database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
        }
        else {
            std::cerr << "Error getting the database.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 6. Get tables.
    Aws::String tableName;
    {
        Aws::Glue::Model::GetTablesRequest request;
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
```

```
        std::vector<Aws::Glue::Model::Table> all_tables;
        Aws::String nextToken; // Used for pagination.
        do {
            Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

            if (outcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
                all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
                nextToken = outcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error getting the tables. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                             clientConfig);
                return false;
            }
        } while (!nextToken.empty());

        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
    }
```

```
   // 7. Create a job.
   {
       Aws::Glue::Model::CreateJobRequest request;
       request.SetName(JOB_NAME);
       request.SetRole(roleArn);
       request.SetGlueVersion(GLUE_VERSION);

       Aws::Glue::Model::JobCommand command;
       command.SetName(JOB_COMMAND_NAME);
       command.SetPythonVersion(JOB_PYTHON_VERSION);
       command.SetScriptLocation(
               Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
       request.SetCommand(command);

       Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

       if (outcome.IsSuccess()) {
           std::cout << "Successfully created the job." << std::endl;
       }
       else {
           std::cerr << "Error creating the job. " <<
outcome.GetError().GetMessage()
                     << std::endl;
           deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                        clientConfig);
           return false;
       }
   }

   // 8. Start a job run.
   {
       Aws::Glue::Model::StartJobRunRequest request;
       request.SetJobName(JOB_NAME);

       Aws::Map<Aws::String, Aws::String> arguments;
       arguments["--input_database"] = CRAWLER_DATABASE_NAME;
       arguments["--input_table"] = tableName;
       arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
       request.SetArguments(arguments);

       Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);
```

```
        if (outcome.IsSuccess()) {
            std::cout << "Successfully started the job." << std::endl;

            Aws::String jobRunId = outcome.GetResult().GetJobRunId();

            int iterator = 0;
            bool done = false;
            while (!done) {
                ++iterator;
                std::this_thread::sleep_for(std::chrono::seconds(1));
                Aws::Glue::Model::GetJobRunRequest jobRunRequest;
                jobRunRequest.SetJobName(JOB_NAME);
                jobRunRequest.SetRunId(jobRunId);

                Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                        jobRunRequest);

                if (jobRunOutcome.IsSuccess()) {
                    const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                    Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();

                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
```

```
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                            jobRunState) <<
                                    ". " << iterator <<
                                    " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
                              << jobRunOutcome.GetError().GetMessage()
                              << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
    }

    // 9. List the output data stored in the S3 bucket.
    {
        Aws::S3::S3Client s3Client;
        Aws::S3::Model::ListObjectsV2Request request;
        request.SetBucket(bucketName);
        request.SetPrefix(OUTPUT_FILE_PREFIX);

        Aws::String continuationToken; // Used for pagination.
        std::vector<Aws::S3::Model::Object> allObjects;
        do {
            if (!continuationToken.empty()) {
                request.SetContinuationToken(continuationToken);
```

```
            }
            Aws::S3::Model::ListObjectsV2Outcome outcome =
 s3Client.ListObjectsV2(
                    request);

            if (outcome.IsSuccess()) {
                const std::vector<Aws::S3::Model::Object> &objects =
                        outcome.GetResult().GetContents();
                allObjects.insert(allObjects.end(), objects.begin(),
 objects.end());
                continuationToken =
 outcome.GetResult().GetNextContinuationToken();
            }
            else {
                std::cerr << "Error listing objects. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!continuationToken.empty());

        std::cout << "Data from your job is in " << allObjects.size() <<
                  " files in the S3 bucket, " << bucketName << "." << std::endl;

        for (size_t i = 0; i < allObjects.size(); ++i) {
            std::cout << "    " << i + 1 << ". " << allObjects[i].GetKey()
                      << std::endl;
        }

        int objectIndex = askQuestionForIntRange(
                std::string(
                        "Enter the number of a block to download it and see the
 first ") +
                std::to_string(LINES_OF_RUN_FILE_TO_DISPLAY) +
                " lines of JSON output in the block: ", 1,
                static_cast<int>(allObjects.size()));

        Aws::String objectKey = allObjects[objectIndex - 1].GetKey();

        std::stringstream stringStream;
        if (getObjectFromBucket(bucketName, objectKey, stringStream,
                                clientConfig)) {
            for (int i = 0; i < LINES_OF_RUN_FILE_TO_DISPLAY && stringStream; +
+i) {
```

```
                std::string line;
                std::getline(stringStream, line);
                std::cout << "    " << line << std::endl;
            }
        }
        else {
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
    }

    // 10. List all the jobs.
    Aws::String jobName;
    {
        Aws::Glue::Model::ListJobsRequest listJobsRequest;

        Aws::String nextToken;
        std::vector<Aws::String> allJobNames;

        do {
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
            }
        } while (!nextToken.empty());
        std::cout << "Your account has " << allJobNames.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < allJobNames.size(); ++i) {
```

```
            std::cout << "   " << i + 1 << ". " << allJobNames[i] << std::endl;
        }
        int jobIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobNames.size()) +
                " to see the list of runs for a job: ",
                1, static_cast<int>(allJobNames.size()));

        jobName = allJobNames[jobIndex - 1];
    }

    // 11. Get the job runs for a job.
    Aws::String jobRunID;
    if (!jobName.empty()) {
        Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
        getJobRunsRequest.SetJobName(jobName);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::Glue::Model::JobRun> allJobRuns;
        do {
            if (!nextToken.empty()) {
                getJobRunsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                    getJobRunsRequest);

            if (jobRunsOutcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
                allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

                nextToken = jobRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error getting job runs. "
                          << jobRunsOutcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "There are " << allJobRuns.size() << " runs in the job '"
```

```
                    <<
                    jobName << "'." << std::endl;

        for (size_t i = 0; i < allJobRuns.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << allJobRuns[i].GetJobName()
                      << std::endl;
        }

        int runIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobRuns.size()) +
                " to see details for a run: ",
                1, static_cast<int>(allJobRuns.size()));
        jobRunID = allJobRuns[runIndex - 1].GetId();
    }

    // 12. Get a single job run.
    if (!jobRunID.empty()) {
        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
 jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
    }

    return deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
 bucketName,
                        clientConfig);
}
```

```
//! Cleanup routine to delete created assets.
/*!
 \\sa deleteAssets()
 \param crawler: Name of an AWS Glue crawler.
 \param database: The name of an AWS Glue database.
 \param job: The name of an AWS Glue job.
 \param bucketName: The name of an S3 bucket.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAssets(const Aws::String &crawler, const Aws::String
 &database,
                                const Aws::String &job, const Aws::String
 &bucketName,
                                const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::Glue::GlueClient client(clientConfig);
    bool result = true;

    // 13. Delete a job.
    if (!job.empty()) {
        Aws::Glue::Model::DeleteJobRequest request;
        request.SetJobName(job);

        Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the job." << std::endl;
        }
        else {
            std::cerr << "Error deleting the job. " <<
 outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
    }

    // 14. Delete a database.
    if (!database.empty()) {
        Aws::Glue::Model::DeleteDatabaseRequest request;
        request.SetName(database);

        Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
```

```
                request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the database." << std::endl;
        }
        else {
            std::cerr << "Error deleting database. " <<
 outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
    }

    // 15. Delete a crawler.
    if (!crawler.empty()) {
        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
 client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
    }

    // 16. Delete the job script and run data from the S3 bucket.
    result &= AwsDoc::Glue::deleteAllObjectsInS3Bucket(bucketName,
                                                       clientConfig);
    return result;
}

//! Routine which uploads a file to an S3 bucket.
/*!
 \\sa uploadFile()
 \param bucketName: An S3 bucket created in the setup.
 \param filePath: The path of the file to upload.
 \param fileName The name for the uploaded file.
 \param clientConfig: AWS client configuration.
```

```
 \return bool: Successful completion.
 */
bool
AwsDoc::Glue::uploadFile(const Aws::String &bucketName,
                         const Aws::String &filePath,
                         const Aws::String &fileName,
                         const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(fileName);

    std::shared_ptr<Aws::IOStream> inputData =
            Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                          filePath.c_str(),
                                          std::ios_base::in |
 std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << filePath << std::endl;
        return false;
    }

    request.SetBody(inputData);

    Aws::S3::Model::PutObjectOutcome outcome =
            s3_client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Added object '" << filePath << "' to bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which deletes all objects in an S3 bucket.
/*!
 \\sa deleteAllObjectsInS3Bucket()
```

```
 \param bucketName: The S3 bucket name.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAllObjectsInS3Bucket(const Aws::String &bucketName,
                                              const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::ListObjectsV2Request listObjectsRequest;
    listObjectsRequest.SetBucket(bucketName);

    Aws::String continuationToken; // Used for pagination.
    bool result = true;
    do {
        if (!continuationToken.empty()) {
            listObjectsRequest.SetContinuationToken(continuationToken);
        }

        Aws::S3::Model::ListObjectsV2Outcome listObjectsOutcome =
 client.ListObjectsV2(
                listObjectsRequest);

        if (listObjectsOutcome.IsSuccess()) {
            const std::vector<Aws::S3::Model::Object> &objects =
 listObjectsOutcome.GetResult().GetContents();
            if (!objects.empty()) {
                Aws::S3::Model::DeleteObjectsRequest deleteObjectsRequest;
                deleteObjectsRequest.SetBucket(bucketName);

                std::vector<Aws::S3::Model::ObjectIdentifier> objectIdentifiers;
                for (const Aws::S3::Model::Object &object: objects) {
                    objectIdentifiers.push_back(
                            Aws::S3::Model::ObjectIdentifier().WithKey(
                                    object.GetKey()));
                }
                Aws::S3::Model::Delete objectsDelete;
                objectsDelete.SetObjects(objectIdentifiers);
                objectsDelete.SetQuiet(true);
                deleteObjectsRequest.SetDelete(objectsDelete);

                Aws::S3::Model::DeleteObjectsOutcome deleteObjectsOutcome =
                        client.DeleteObjects(deleteObjectsRequest);

                if (!deleteObjectsOutcome.IsSuccess()) {
```

```
                    std::cerr << "Error deleting objects. " <<
                              deleteObjectsOutcome.GetError().GetMessage() <<
 std::endl;
                    result = false;
                    break;
                }
                else {
                    std::cout << "Successfully deleted the objects." <<
 std::endl;

                }
            }
            else {
                std::cout << "No objects to delete in '" << bucketName << "'."
                          << std::endl;
            }

            continuationToken =
 listObjectsOutcome.GetResult().GetNextContinuationToken();
        }
        else {
            std::cerr << "Error listing objects. "
                      << listObjectsOutcome.GetError().GetMessage() << std::endl;
            result = false;
            break;
        }
    } while (!continuationToken.empty());

    return result;
}

//! Routine which retrieves an object from an S3 bucket.
/*!
 \\sa getObjectFromBucket()
 \param bucketName: The S3 bucket name.
 \param objectKey: The object's name.
 \param objectStream: A stream to receive the retrieved data.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::getObjectFromBucket(const Aws::String &bucketName,
                                       const Aws::String &objectKey,
                                       std::ostream &objectStream,
```

```
                                        const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome = client.GetObject(request);


    if (outcome.IsSuccess()) {
        std::cout << "Successfully retrieved '" << objectKey << "'." <<
 std::endl;
        auto &body = outcome.GetResult().GetBody();
        objectStream << body.rdbuf();
    }
    else {
        std::cerr << "Error retrieving object. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für C++ -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun

- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * To set up the resources, see this documentation topic:
 *
 * https://docs.aws.amazon.com/glue/latest/ug/tutorial-add-crawler.html
 *
 * This example performs the following tasks:
 *
 * 1. Create a database.
 * 2. Create a crawler.
 * 3. Get a crawler.
 * 4. Start a crawler.
 * 5. Get a database.
 * 6. Get tables.
 * 7. Create a job.
 * 8. Start a job run.
```

```
 * 9. List all jobs.
 * 10. Get job runs.
 * 11. Delete a job.
 * 12. Delete a database.
 * 13. Delete a crawler.
 */

public class GlueScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException {
        final String usage = """

            Usage:
                <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>
 <scriptLocation> <locationUri> <bucketNameSc>\s

            Where:
                iam - The ARN of the IAM role that has AWS Glue and S3
 permissions.\s
                s3Path - The Amazon Simple Storage Service (Amazon S3) target
 that contains data (for example, s3://<bucket name>/read).
                cron - A cron expression used to specify the schedule  (i.e.,
 cron(15 12 * * ? *).
                dbName - The database name.\s
                crawlerName - The name of the crawler.\s
                jobName - The name you assign to this job definition.
                scriptLocation - The Amazon S3 path to a script that runs a job.
                locationUri - The location of the database (you can find this
 file in resources folder).
                bucketNameSc - The Amazon S3 bucket name used when creating a job
                """;

        if (args.length != 9) {
            System.out.println(usage);
            return;
        }
        Scanner scanner = new Scanner(System.in);
        String iam = args[0];
        String s3Path = args[1];
        String cron = args[2];
        String dbName = args[3];
        String crawlerName = args[4];
```

```
        String jobName = args[5];
        String scriptLocation = args[6];
        String locationUri = args[7];
        String bucketNameSc = args[8];

        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
            .region(region)
            .build();
        System.out.println(DASHES);
        System.out.println("Welcome to the AWS Glue scenario.");
        System.out.println("""
            AWS Glue is a fully managed extract, transform, and load (ETL)
service provided by Amazon
            Web Services (AWS). It is designed to simplify the process of
building, running, and maintaining
            ETL pipelines, which are essential for data integration and data
warehousing tasks.

            One of the key features of AWS Glue is its ability to automatically
discover and catalog data
            stored in various sources, such as Amazon S3, Amazon RDS, Amazon
Redshift, and other databases.
            This cataloging process creates a central metadata repository, known
as the AWS Glue Data Catalog,
            which provides a unified view of an organization's data assets. This
metadata can then be used to
            create ETL jobs, which can be scheduled and run on-demand or on a
regular basis.

            Lets get started.

            """);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a database.");
        try {
            createDatabase(glueClient, dbName, locationUri);
        } catch (GlueException e) {
            if (e.awsErrorDetails().errorMessage().equals("Database already
exists.")) {
```

```
                System.out.println("Database " + dbName + " already exists.
Skipping creation.");
            } else {
                System.err.println(e.awsErrorDetails().errorMessage());
                return;
            }
        }

        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Create a crawler.");
        try {
            createGlueCrawler(glueClient, iam, s3Path, cron, dbName,
crawlerName);
        } catch (GlueException e) {
            if (e.awsErrorDetails().errorMessage().contains("already exists")) {
                System.out.println("Crawler " + crawlerName + " already exists.
Skipping creation.");
            } else {
                System.err.println(e.awsErrorDetails().errorMessage());
                System.exit(1);
            }
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Get a crawler.");
        try {
            getSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Start a crawler.");
        try {
            startSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Get a database.");
        try {
            getSpecificDatabase(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait 5 min for the tables to become available");
        TimeUnit.MINUTES.sleep(5);
        System.out.println("6. Get tables.");
        String myTableName;
        try {
            myTableName = getGlueTables(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Create a job.");
        try {
            createJob(glueClient, jobName, iam, scriptLocation);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Start a Job run.");
```

```
        try {
            startJob(glueClient, jobName, dbName, myTableName, bucketNameSc);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. List all jobs.");
        try {
            getAllJobs(glueClient);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Get job runs.");
        try {
            getJobRuns(glueClient, jobName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Delete a job.");
        try {
            deleteJob(glueClient, jobName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        System.out.println("*** Wait 5 MIN for the " + crawlerName + " to stop");
        TimeUnit.MINUTES.sleep(5);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("12. Delete a database.");
        try {
            deleteDatabase(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Delete a crawler.");
        try {
            deleteSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Successfully completed the AWS Glue Scenario");
        System.out.println(DASHES);
    }


    /**
     * Creates a Glue database with the specified name and location URI.
     *
     * @param glueClient  The Glue client to use for the database creation.
     * @param dbName      The name of the database to create.
     * @param locationUri The location URI for the database.
     */
    public static void createDatabase(GlueClient glueClient, String dbName,
String locationUri) {
        try {
            DatabaseInput input = DatabaseInput.builder()
                .description("Built with the AWS SDK for Java V2")
                .name(dbName)
                .locationUri(locationUri)
                .build();
```

```
            CreateDatabaseRequest request = CreateDatabaseRequest.builder()
                .databaseInput(input)
                .build();

            glueClient.createDatabase(request);
            System.out.println(dbName + " was successfully created");

        } catch (GlueException e) {
            throw e;
        }
    }


    /**
     * Creates a new AWS Glue crawler using the AWS Glue Java API.
     *
     * @param glueClient  the AWS Glue client used to interact with the AWS Glue
service
     * @param iam         the IAM role that the crawler will use to access the
data source
     * @param s3Path      the S3 path that the crawler will scan for data
     * @param cron        the cron expression that defines the crawler's schedule
     * @param dbName      the name of the AWS Glue database where the crawler
will store the metadata
     * @param crawlerName the name of the crawler to be created
     */
    public static void createGlueCrawler(GlueClient glueClient,
                                         String iam,
                                         String s3Path,
                                         String cron,
                                         String dbName,
                                         String crawlerName) {

        try {
            S3Target s3Target = S3Target.builder()
                .path(s3Path)
                .build();

            List<S3Target> targetList = new ArrayList<>();
            targetList.add(s3Target);
            CrawlerTargets targets = CrawlerTargets.builder()
                .s3Targets(targetList)
                .build();
```

```
            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                .databaseName(dbName)
                .name(crawlerName)
                .description("Created by the AWS Glue Java API")
                .targets(targets)
                .role(iam)
                .schedule(cron)
                .build();

            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Retrieves a specific crawler from the AWS Glue service and waits for it to
be in the "READY" state.
     *
     * @param glueClient  the AWS Glue client used to interact with the Glue
service
     * @param crawlerName the name of the crawler to be retrieved
     */
    public static void getSpecificCrawler(GlueClient glueClient, String
crawlerName) throws InterruptedException {
        try {
            GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            boolean ready = false;
            while (!ready) {
                GetCrawlerResponse response =
glueClient.getCrawler(crawlerRequest);
                String status = response.crawler().stateAsString();
                if (status.compareTo("READY") == 0) {
                    ready = true;
                }
                Thread.sleep(3000);
            }

            System.out.println("The crawler is now ready");
```

```
        } catch (GlueException | InterruptedException e) {
            throw e;
        }
    }

    /**
     * Starts a specific AWS Glue crawler.
     *
     * @param glueClient  the AWS Glue client to use for the crawler operation
     * @param crawlerName the name of the crawler to start
     * @throws GlueException if there is an error starting the crawler
     */
    public static void startSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
            StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            glueClient.startCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully started!");

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Retrieves the specific database from the AWS Glue service.
     *
     * @param glueClient   an instance of the AWS Glue client used to interact
with the service
     * @param databaseName the name of the database to retrieve
     * @throws GlueException if there is an error retrieving the database from
the AWS Glue service
     */
    public static void getSpecificDatabase(GlueClient glueClient, String
databaseName) {
        try {
            GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
                .name(databaseName)
                .build();
```

```
            GetDatabaseResponse response =
glueClient.getDatabase(databasesRequest);
            Instant createDate = response.database().createTime();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                .withLocale(Locale.US)
                .withZone(ZoneId.systemDefault());

            formatter.format(createDate);
            System.out.println("The create date of the database is " +
createDate);

        } catch (GlueException e) {
            throw e;
        }
    }


    /**
     * Retrieves the names of the tables in the specified Glue database.
     *
     * @param glueClient the Glue client to use for the operation
     * @param dbName     the name of the Glue database to retrieve the table
names from
     * @return the name of the first table retrieved, or an empty string if no
tables were found
     */
    public static String getGlueTables(GlueClient glueClient, String dbName) {
        String myTableName = "";
        try {
            GetTablesRequest tableRequest = GetTablesRequest.builder()
                .databaseName(dbName)
                .build();

            GetTablesResponse response = glueClient.getTables(tableRequest);
            List<Table> tables = response.tableList();
            if (tables.isEmpty()) {
                System.out.println("No tables were returned");
            } else {
                for (Table table : tables) {
                    myTableName = table.name();
                    System.out.println("Table name is: " + myTableName);
```

```
                }
            }

        } catch (GlueException e) {
            throw e;
        }
        return myTableName;
    }


    /**
     * Starts a job run in AWS Glue.
     *
     * @param glueClient    the AWS Glue client to use for the job run
     * @param jobName       the name of the Glue job to run
     * @param inputDatabase the name of the input database
     * @param inputTable    the name of the input table
     * @param outBucket     the URL of the output S3 bucket
     * @throws GlueException if there is an error starting the job run
     */
    public static void startJob(GlueClient glueClient, String jobName, String
inputDatabase, String inputTable,
                                String outBucket) {
        try {
            Map<String, String> myMap = new HashMap<>();
            myMap.put("--input_database", inputDatabase);
            myMap.put("--input_table", inputTable);
            myMap.put("--output_bucket_url", outBucket);

            StartJobRunRequest runRequest = StartJobRunRequest.builder()
                .workerType(WorkerType.G_1_X)
                .numberOfWorkers(10)
                .arguments(myMap)
                .jobName(jobName)
                .build();

            StartJobRunResponse response = glueClient.startJobRun(runRequest);
            System.out.println("The request Id of the job is " +
response.responseMetadata().requestId());

        } catch (GlueException e) {
            throw e;
        }
    }
```

```
   /**
    * Creates a new AWS Glue job.
    *
    * @param glueClient     the AWS Glue client to use for the operation
    * @param jobName        the name of the job to create
    * @param iam            the IAM role to associate with the job
    * @param scriptLocation the location of the script to be used by the job
    * @throws GlueException if there is an error creating the job
    */
   public static void createJob(GlueClient glueClient, String jobName, String
iam, String scriptLocation) {
       try {
           JobCommand command = JobCommand.builder()
               .pythonVersion("3")
               .name("glueetl")
               .scriptLocation(scriptLocation)
               .build();

           CreateJobRequest jobRequest = CreateJobRequest.builder()
               .description("A Job created by using the AWS SDK for Java V2")
               .glueVersion("2.0")
               .workerType(WorkerType.G_1_X)
               .numberOfWorkers(10)
               .name(jobName)
               .role(iam)
               .command(command)
               .build();

           glueClient.createJob(jobRequest);
           System.out.println(jobName + " was successfully created.");

       } catch (GlueException e) {
           throw e;
       }
   }


   /**
    * Retrieves and prints information about all the jobs in the Glue data
catalog.
    *
```

```
     * @param glueClient the Glue client used to interact with the AWS Glue
service
     */
    public static void getAllJobs(GlueClient glueClient) {
        try {
            GetJobsRequest jobsRequest = GetJobsRequest.builder()
                .maxResults(10)
                .build();

            GetJobsResponse jobsResponse = glueClient.getJobs(jobsRequest);
            List<Job> jobs = jobsResponse.jobs();
            for (Job job : jobs) {
                System.out.println("Job name is : " + job.name());
                System.out.println("The job worker type is : " +
job.workerType().name());
            }

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Retrieves the job runs for a given Glue job and prints the status of the
job runs.
     *
     * @param glueClient the Glue client used to make API calls
     * @param jobName    the name of the Glue job to retrieve the job runs for
     */
    public static void getJobRuns(GlueClient glueClient, String jobName) {
        try {
            GetJobRunsRequest runsRequest = GetJobRunsRequest.builder()
                .jobName(jobName)
                .maxResults(20)
                .build();

            boolean jobDone = false;
            while (!jobDone) {
                GetJobRunsResponse response = glueClient.getJobRuns(runsRequest);
                List<JobRun> jobRuns = response.jobRuns();
                for (JobRun jobRun : jobRuns) {
                    String jobState = jobRun.jobRunState().name();
                    if (jobState.compareTo("SUCCEEDED") == 0) {
                        System.out.println(jobName + " has succeeded");
```

```
                        jobDone = true;

                    } else if (jobState.compareTo("STOPPED") == 0) {
                        System.out.println("Job run has stopped");
                        jobDone = true;

                    } else if (jobState.compareTo("FAILED") == 0) {
                        System.out.println("Job run has failed");
                        jobDone = true;

                    } else if (jobState.compareTo("TIMEOUT") == 0) {
                        System.out.println("Job run has timed out");
                        jobDone = true;

                    } else {
                        System.out.println("*** Job run state is " +
jobRun.jobRunState().name());
                        System.out.println("Job run Id is " + jobRun.id());
                        System.out.println("The Glue version is " +
jobRun.glueVersion());
                    }
                    TimeUnit.SECONDS.sleep(5);
                }
            }

        } catch (GlueException e) {
            throw e;
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }


    /**
     * Deletes a Glue job.
     *
     * @param glueClient the Glue client to use for the operation
     * @param jobName    the name of the job to be deleted
     * @throws GlueException if there is an error deleting the job
     */
    public static void deleteJob(GlueClient glueClient, String jobName) {
        try {
            DeleteJobRequest jobRequest = DeleteJobRequest.builder()
                .jobName(jobName)
```

```
                .build();

            glueClient.deleteJob(jobRequest);
            System.out.println(jobName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Deletes a AWS Glue Database.
     *
     * @param glueClient   An instance of the AWS Glue client used to interact
with the AWS Glue service.
     * @param databaseName The name of the database to be deleted.
     * @throws GlueException If an error occurs while deleting the database.
     */
    public static void deleteDatabase(GlueClient glueClient, String databaseName)
{
        try {
            DeleteDatabaseRequest request = DeleteDatabaseRequest.builder()
                .name(databaseName)
                .build();

            glueClient.deleteDatabase(request);
            System.out.println(databaseName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
    }


    /**
     * Deletes a specific AWS Glue crawler.
     *
     * @param glueClient  the AWS Glue client object
     * @param crawlerName the name of the crawler to be deleted
     * @throws GlueException if an error occurs during the deletion process
     */
    public static void deleteSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
```

```
            DeleteCrawlerRequest deleteCrawlerRequest =
 DeleteCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            glueClient.deleteCrawler(deleteCrawlerRequest);
            System.out.println(crawlerName + " was deleted");

        } catch (GlueException e) {
            throw e;
        }
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            System.out.println("");
            System.out.println("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
                System.out.println("Continuing with the program...");
                System.out.println("");
                break;
            } else {
                // Handle invalid input.
                System.out.println("Invalid input. Please try again.");
            }
        }
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK for Java 2.x -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler

- GetDatabase
- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie einen Crawler und führen Sie ihn aus, der einen öffentlichen Amazon Simple Storage Service (Amazon S3)-Bucket crawlt und eine Metadatendatenbank generiert, die die gefundenen CSV-formatierten Daten beschreibt.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});

  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });

  return client.send(command);
```

```
};

const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};

const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};

const crawlerExists = async ({ getCrawler }, crawlerName) => {
  try {
    await getCrawler(crawlerName);
    return true;
  } catch {
    return false;
  }
};

/**
 * @param {{ createCrawler: import('../../../actions/create-
crawler.js').createCrawler}} actions
 */
const makeCreateCrawlerStep = (actions) => async (context) => {
  if (await crawlerExists(actions, process.env.CRAWLER_NAME)) {
    log("Crawler already exists. Skipping creation.");
  } else {
    await actions.createCrawler(
      process.env.CRAWLER_NAME,
      process.env.ROLE_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_PREFIX,
```

```
      process.env.S3_TARGET_PATH,
    );

    log("Crawler created successfully.", { type: "success" });
  }

  return { ...context };
};

/**
 * @param {(name: string) => Promise<import('@aws-sdk/client-
glue').GetCrawlerCommandOutput>} getCrawler
 * @param {string} crawlerName
 */
const waitForCrawler = async (getCrawler, crawlerName) => {
  const waitTimeInSeconds = 30;
  const { Crawler } = await getCrawler(crawlerName);

  if (!Crawler) {
    throw new Error(`Crawler with name ${crawlerName} not found.`);
  }

  if (Crawler.State === "READY") {
    return;
  }

  log(`Crawler is ${Crawler.State}. Waiting ${waitTimeInSeconds} seconds...`);
  await wait(waitTimeInSeconds);
  return waitForCrawler(getCrawler, crawlerName);
};

const makeStartCrawlerStep =
  ({ startCrawler, getCrawler }) =>
  async (context) => {
    log("Starting crawler.");
    await startCrawler(process.env.CRAWLER_NAME);
    log("Crawler started.", { type: "success" });

    log("Waiting for crawler to finish running. This can take a while.");
    await waitForCrawler(getCrawler, process.env.CRAWLER_NAME);
    log("Crawler ready.", { type: "success" });

    return { ...context };
  };
```

Listen Sie Informationen zu Datenbanken und Tabellen in Ihrem auf AWS Glue Data Catalog.

```
const getDatabase = (name) => {
  const client = new GlueClient({});

  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};

const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};

const makeGetDatabaseStep =
  ({ getDatabase }) =>
  async (context) => {
    const {
      Database: { Name },
    } = await getDatabase(process.env.DATABASE_NAME);
    log(`Database: ${Name}`);
    return { ...context };
  };

/**
 * @param {{ getTables: () => Promise<import('@aws-sdk/client-
glue').GetTablesCommandOutput}} config
 */
const makeGetTablesStep =
  ({ getTables }) =>
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME);
    log("Tables:");
```

```
    log(TableList.map((table) => `  • ${table.Name}\n`));
    return { ...context };
  };
```

Erstellen und führen Sie einen Auftrag aus, der CSV-Daten aus dem Amazon-S3-Quell-Bucket extrahiert, sie durch Entfernen und Umbenennen von Feldern transformiert und die JSON-formatierte Ausgabe in einen anderen Amazon-S3-Bucket lädt.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
};

const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};

const makeCreateJobStep =
  ({ createJob }) =>
```

```
  async (context) => {
    log("Creating Job.");
    await createJob(
      process.env.JOB_NAME,
      process.env.ROLE_NAME,
      process.env.BUCKET_NAME,
      process.env.PYTHON_SCRIPT_KEY,
    );
    log("Job created.", { type: "success" });

    return { ...context };
  };

/**
 * @param {(name: string, runId: string) => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput> }  getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
 */
const waitForJobRun = async (getJobRun, jobName, jobRunId) => {
  const waitTimeInSeconds = 30;
  const { JobRun } = await getJobRun(jobName, jobRunId);

  if (!JobRun) {
    throw new Error(`Job run with id ${jobRunId} not found.`);
  }

  switch (JobRun.JobRunState) {
    case "FAILED":
    case "TIMEOUT":
    case "STOPPED":
    case "ERROR":
      throw new Error(
        `Job ${JobRun.JobRunState}. Error: ${JobRun.ErrorMessage}`,
      );
    case "SUCCEEDED":
      return;
    default:
      break;
  }

  log(
    `Job ${JobRun.JobRunState}. Waiting ${waitTimeInSeconds} more seconds...`,
  );
```

```
  await wait(waitTimeInSeconds);
  return waitForJobRun(getJobRun, jobName, jobRunId);
};

/**
 * @param {{ prompter: { prompt: () => Promise<{ shouldOpen: boolean }>} }}
 context
 */
const promptToOpen = async (context) => {
  const { shouldOpen } = await context.prompter.prompt({
    name: "shouldOpen",
    type: "confirm",
    message: "Open the output bucket in your browser?",
  });

  if (shouldOpen) {
    return open(
      `https://s3.console.aws.amazon.com/s3/buckets/${process.env.BUCKET_NAME} to
 view the output.`,
    );
  }
};

const makeStartJobRunStep =
  ({ startJobRun, getJobRun }) =>
  async (context) => {
    log("Starting job.");
    const { JobRunId } = await startJobRun(
      process.env.JOB_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_NAME,
      process.env.BUCKET_NAME,
    );
    log("Job started.", { type: "success" });

    log("Waiting for job to finish running. This can take a while.");
    await waitForJobRun(getJobRun, process.env.JOB_NAME, JobRunId);
    log("Job run succeeded.", { type: "success" });

    await promptToOpen(context);

    return { ...context };
  };
```

Listet Informationen über Auftragsausführungen auf und zeigt einige der transformierten Daten an.

```
const getJobRuns = (jobName) => {
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};

/**
 * @typedef {{ prompter: { prompt: () => Promise<{jobName: string}> } }} Context
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput>} getJobRun
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunsCommandOutput} getJobRuns
 */

/**
 *
 * @param {getJobRun} getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
```

```
 */
const logJobRunDetails = async (getJobRun, jobName, jobRunId) => {
  const { JobRun } = await getJobRun(jobName, jobRunId);
  log(JobRun, { type: "object" });
};

/**
 *
 * @param {{getJobRuns: getJobRuns, getJobRun: getJobRun }} funcs
 */
const makePickJobRunStep =
  ({ getJobRuns, getJobRun }) =>
  async (/** @type { Context } */ context) => {
    if (context.selectedJobName) {
      const { JobRuns } = await getJobRuns(context.selectedJobName);

      const { jobRunId } = await context.prompter.prompt({
        name: "jobRunId",
        type: "list",
        message: "Select a job run to see details.",
        choices: JobRuns.map((run) => run.Id),
      });

      logJobRunDetails(getJobRun, context.selectedJobName, jobRunId);
    }

    return { ...context };
  };
```

Löscht alle Ressourcen, die von der Demo erstellt wurden.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const deleteTable = (databaseName, tableName) => {
```

```
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
  });

  return client.send(command);
};

const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};

const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });

  return client.send(command);
};

/**
 *
 * @param {import('../../../actions/delete-job.js').deleteJob} deleteJobFn
 * @param {string[]} jobNames
 * @param {{ prompter: { prompt: () => Promise<any> }}} context
 */
const handleDeleteJobs = async (deleteJobFn, jobNames, context) => {
  /**
   * @type {{ selectedJobNames: string[] }}
   */
  const { selectedJobNames } = await context.prompter.prompt({
    name: "selectedJobNames",
    type: "checkbox",
    message: "Let's clean up jobs. Select jobs to delete.",
```

```
    choices: jobNames,
  });

  if (selectedJobNames.length === 0) {
    log("No jobs selected.");
  } else {
    log("Deleting jobs.");
    await Promise.all(
      selectedJobNames.map((n) => deleteJobFn(n).catch(console.error)),
    );
    log("Jobs deleted.", { type: "success" });
  }
};

/**
 * @param {{
 *   listJobs: import('../../../actions/list-jobs.js').listJobs,
 *   deleteJob: import('../../../actions/delete-job.js').deleteJob
 * }} config
 */
const makeCleanUpJobsStep =
  ({ listJobs, deleteJob }) =>
  async (context) => {
    const { JobNames } = await listJobs();
    if (JobNames.length > 0) {
      await handleDeleteJobs(deleteJob, JobNames, context);
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-table.js').deleteTable} deleteTable
 * @param {string} databaseName
 * @param {string[]} tableNames
 */
const deleteTables = (deleteTable, databaseName, tableNames) =>
  Promise.all(
    tableNames.map((tableName) =>
      deleteTable(databaseName, tableName).catch(console.error),
    ),
  );

/**
```

```
 * @param {{
 *  getTables: import('../../../actions/get-tables.js').getTables,
 *  deleteTable: import('../../../actions/delete-table.js').deleteTable
 * }} config
 */
const makeCleanUpTablesStep =
  ({ getTables, deleteTable }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}}} context
   */
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME).catch(
      () => ({ TableList: null }),
    );

    if (TableList && TableList.length > 0) {
      /**
       * @type {{ tableNames: string[] }}
       */
      const { tableNames } = await context.prompter.prompt({
        name: "tableNames",
        type: "checkbox",
        message: "Let's clean up tables. Select tables to delete.",
        choices: TableList.map((t) => t.Name),
      });

      if (tableNames.length === 0) {
        log("No tables selected.");
      } else {
        log("Deleting tables.");
        await deleteTables(deleteTable, process.env.DATABASE_NAME, tableNames);
        log("Tables deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-database.js').deleteDatabase}
 deleteDatabase
 * @param {string[]} databaseNames
 */
const deleteDatabases = (deleteDatabase, databaseNames) =>
```

```
  Promise.all(
    databaseNames.map((dbName) => deleteDatabase(dbName).catch(console.error)),
  );

/**
 * @param {{
 *   getDatabases: import('../../../actions/get-databases.js').getDatabases
 *   deleteDatabase: import('../../../actions/delete-database.js').deleteDatabase
 * }} config
 */
const makeCleanUpDatabasesStep =
  ({ getDatabases, deleteDatabase }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}} context
   */
  async (context) => {
    const { DatabaseList } = await getDatabases();

    if (DatabaseList.length > 0) {
      /** @type {{ dbNames: string[] }} */
      const { dbNames } = await context.prompter.prompt({
        name: "dbNames",
        type: "checkbox",
        message: "Let's clean up databases. Select databases to delete.",
        choices: DatabaseList.map((db) => db.Name),
      });

      if (dbNames.length === 0) {
        log("No databases selected.");
      } else {
        log("Deleting databases.");
        await deleteDatabases(deleteDatabase, dbNames);
        log("Databases deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

const cleanUpCrawlerStep = async (context) => {
  log("Deleting crawler.");

  try {
    await deleteCrawler(process.env.CRAWLER_NAME);
```

```
    log("Crawler deleted.", { type: "success" });
  } catch (err) {
    if (err.name === "EntityNotFoundException") {
      log("Crawler is already deleted.");
    } else {
      throw err;
    }
  }

  return { ...context };
};
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für JavaScript -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>
 <scriptLocation> <locationUri>

        Where:
            iam - The Amazon Resource Name (ARN) of the AWS Identity and Access
 Management (IAM) role that has AWS Glue and Amazon Simple Storage Service
 (Amazon S3) permissions.
            s3Path - The Amazon Simple Storage Service (Amazon S3) target that
 contains data (for example, CSV data).
            cron - A cron expression used to specify the schedule (for example,
 cron(15 12 * * ? *).
            dbName - The database name.
            crawlerName - The name of the crawler.
            jobName - The name you assign to this job definition.
            scriptLocation - Specifies the Amazon S3 path to a script that runs a
 job.
            locationUri - Specifies the location of the database
        """

    if (args.size != 8) {
        println(usage)
        exitProcess(1)
    }

    val iam = args[0]
    val s3Path = args[1]
    val cron = args[2]
    val dbName = args[3]
    val crawlerName = args[4]
```

```
    val jobName = args[5]
    val scriptLocation = args[6]
    val locationUri = args[7]

    println("About to start the AWS Glue Scenario")
    createDatabase(dbName, locationUri)
    createCrawler(iam, s3Path, cron, dbName, crawlerName)
    getCrawler(crawlerName)
    startCrawler(crawlerName)
    getDatabase(dbName)
    getGlueTables(dbName)
    createJob(jobName, iam, scriptLocation)
    startJob(jobName)
    getJobs()
    getJobRuns(jobName)
    deleteJob(jobName)
    println("*** Wait for 5 MIN so the $crawlerName is ready to be deleted")
    TimeUnit.MINUTES.sleep(5)
    deleteMyDatabase(dbName)
    deleteCrawler(crawlerName)
}

suspend fun createDatabase(
    dbName: String?,
    locationUriVal: String?,
) {
    val input =
        DatabaseInput {
            description = "Built with the AWS SDK for Kotlin"
            name = dbName
            locationUri = locationUriVal
        }

    val request =
        CreateDatabaseRequest {
            databaseInput = input
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createDatabase(request)
        println("The database was successfully created")
    }
}
```

```
suspend fun createCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    val targetList = ArrayList<S3Target>()
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val crawlerRequest =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
            description = "Created by the AWS Glue Java API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createCrawler(crawlerRequest)
        println("$crawlerName was successfully created")
    }
}

suspend fun getCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
```

```
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}

suspend fun startCrawler(crawlerName: String) {
    val crawlerRequest =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.startCrawler(crawlerRequest)
        println("$crawlerName was successfully started.")
    }
}

suspend fun getDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}

suspend fun getGlueTables(dbName: String?) {
    val tableRequest =
        GetTablesRequest {
            databaseName = dbName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getTables(tableRequest)
        response.tableList?.forEach { tableName ->
            println("Table name is ${tableName.name}")
        }
    }
}
```

```
suspend fun startJob(jobNameVal: String?) {
    val runRequest =
        StartJobRunRequest {
            workerType = WorkerType.G1X
            numberOfWorkers = 10
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.startJobRun(runRequest)
        println("The job run Id is ${response.jobRunId}")
    }
}

suspend fun createJob(
    jobName: String,
    iam: String?,
    scriptLocationVal: String?,
) {
    val commandOb =
        JobCommand {
            pythonVersion = "3"
            name = "MyJob1"
            scriptLocation = scriptLocationVal
        }

    val jobRequest =
        CreateJobRequest {
            description = "A Job created by using the AWS SDK for Java V2"
            glueVersion = "2.0"
            workerType = WorkerType.G1X
            numberOfWorkers = 10
            name = jobName
            role = iam
            command = commandOb
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createJob(jobRequest)
        println("$jobName was successfully created.")
    }
}

suspend fun getJobs() {
```

```
    val request =
        GetJobsRequest {
            maxResults = 10
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobs(request)
        response.jobs?.forEach { job ->
            println("Job name is ${job.name}")
        }
    }
}

suspend fun getJobRuns(jobNameVal: String?) {
    val request =
        GetJobRunsRequest {
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobRuns(request)
        response.jobRuns?.forEach { job ->
            println("Job name is ${job.jobName}")
        }
    }
}

suspend fun deleteJob(jobNameVal: String) {
    val jobRequest =
        DeleteJobRequest {
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteJob(jobRequest)
        println("$jobNameVal was successfully deleted")
    }
}

suspend fun deleteMyDatabase(databaseName: String) {
    val request =
        DeleteDatabaseRequest {
            name = databaseName
        }
```

```
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteDatabase(request)
        println("$databaseName was successfully deleted")
    }
}

suspend fun deleteCrawler(crawlerName: String) {
    val request =
        DeleteCrawlerRequest {
            name = crawlerName
        }
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteCrawler(request)
        println("$crawlerName was deleted")
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Kotlin.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler

- StartJobRun

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
namespace Glue;

use Aws\Glue\GlueClient;
use Aws\S3\S3Client;
use AwsUtilities\AWSServiceClass;
use GuzzleHttp\Psr7\Stream;
use Iam\IAMService;

class GettingStartedWithGlue
{
    public function run()
    {
        echo("\n");
        echo("--------------------------------------\n");
        print("Welcome to the AWS Glue getting started demo using PHP!\n");
        echo("--------------------------------------\n");

        $clientArgs = [
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ];
        $uniqid = uniqid();

        $glueClient = new GlueClient($clientArgs);
        $glueService = new GlueService($glueClient);
        $iamService = new IAMService();
        $crawlerName = "example-crawler-test-" . $uniqid;
```

```
        AWSServiceClass::$waitTime = 5;
        AWSServiceClass::$maxWaitAttempts = 20;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $databaseName = "doc-example-database-$uniqid";
        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);
        $glueService->startCrawler($crawlerName);

        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

        //Upload job script
        $s3client = new S3Client($clientArgs);
        $bucketName = "test-glue-bucket-" . $uniqid;
        $s3client->createBucket([
            'Bucket' => $bucketName,
            'CreateBucketConfiguration' => ['LocationConstraint' => 'us-west-2'],
        ]);

        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'run_job.py',
            'SourceFile' => __DIR__ . '/flight_etl_job_script.py'
        ]);
        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'setup_scenario_getting_started.yaml',
            'SourceFile' => __DIR__ . '/setup_scenario_getting_started.yaml'
        ]);

        $tables = $glueService->getTables($databaseName);

        $jobName = 'test-job-' . $uniqid;
```

```
        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
$scriptLocation);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

        $jobRuns = $glueService->getJobRuns($jobName);

        $objects = $s3client->listObjects([
            'Bucket' => $bucketName,
        ])['Contents'];

        foreach ($objects as $object) {
            echo $object['Key'] . "\n";
        }

        echo "Downloading " . $objects[1]['Key'] . "\n";
        /** @var Stream $downloadObject */
        $downloadObject = $s3client->getObject([
            'Bucket' => $bucketName,
            'Key' => $objects[1]['Key'],
        ])['Body']->getContents();
        echo "Here is the first 1000 characters in the object.";
        echo substr($downloadObject, 0, 1000);

        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
            echo "{$jobsName}\n";
        }

        echo "Delete the job.\n";
        $glueClient->deleteJob([
```

```
            'JobName' => $job['Name'],
        ]);

        echo "Delete the tables.\n";
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }

        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

        $deleteObjects = $s3client->listObjectsV2([
            'Bucket' => $bucketName,
        ]);
        echo "Delete all objects in the bucket.\n";
        $deleteObjects = $s3client->deleteObjects([
            'Bucket' => $bucketName,
            'Delete' => [
                'Objects' => $deleteObjects['Contents'],
            ]
        ]);
        echo "Delete the bucket.\n";
        $s3client->deleteBucket(['Bucket' => $bucketName]);

        echo "This job was brought to you by the number $uniqid\n";
    }
}

namespace Glue;

use Aws\Glue\GlueClient;
use Aws\Result;

use function PHPUnit\Framework\isEmpty;

class GlueService extends \AwsUtilities\AWSServiceClass
{
```

```
    protected GlueClient $glueClient;

    public function __construct($glueClient)
    {
        $this->glueClient = $glueClient;
    }

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
    }

    public function createCrawler($crawlerName, $role, $databaseName, $path):
Result
    {
        return $this->customWaiter(function () use ($crawlerName, $role,
$databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
        });
    }

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }

    public function getDatabase(string $databaseName): Result
    {
```

```
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }

    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
            'DatabaseName' => $databaseName,
        ]);
    }

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
'3', $glueVersion = '3.0'): Result
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }

    public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
    {
        return $this->glueClient->startJobRun([
            'JobName' => $jobName,
            'Arguments' => [
                'input_database' => $databaseName,
                'input_table' => $tables['TableList'][0]['Name'],
                'output_bucket_url' => $outputBucketUrl,
                '--input_database' => $databaseName,
                '--input_table' => $tables['TableList'][0]['Name'],
                '--output_bucket_url' => $outputBucketUrl,
            ],
        ]);
    }
```

```
   public function listJobs($maxResults = null, $nextToken = null, $tags = []):
Result
   {
       $arguments = [];
       if ($maxResults) {
           $arguments['MaxResults'] = $maxResults;
       }
       if ($nextToken) {
           $arguments['NextToken'] = $nextToken;
       }
       if (!empty($tags)) {
           $arguments['Tags'] = $tags;
       }
       return $this->glueClient->listJobs($arguments);
   }

   public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
Result
   {
       $arguments = ['JobName' => $jobName];
       if ($maxResults) {
           $arguments['MaxResults'] = $maxResults;
       }
       if ($nextToken) {
           $arguments['NextToken'] = $nextToken;
       }
       return $this->glueClient->getJobRuns($arguments);
   }

   public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
Result
   {
       return $this->glueClient->getJobRun([
           'JobName' => $jobName,
           'RunId' => $runId,
           'PredecessorsIncluded' => $predecessorsIncluded,
       ]);
   }

   public function deleteJob($jobName)
   {
       return $this->glueClient->deleteJob([
           'JobName' => $jobName,
```

```
        ]);
    }

    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }

    public function deleteCrawler($crawlerName)
    {
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
}
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für PHP -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun

- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie eine Klasse, die die im Szenario verwendeten AWS Glue Funktionen umschließt.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
            crawler = response["Crawler"]
        except ClientError as err:
```

```
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return crawler


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
data
        in the target.

        :param name: The name of the crawler.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
Access
                         Management (IAM) role that grants permission to let AWS
Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
crawler.
        :param db_prefix: The prefix to give any database tables that are created
by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
                          the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
        except ClientError as err:
            logger.error(
```

```
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
```

```
            return response["Database"]


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["TableList"]


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
that can
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
run as
                                part of the job. The script defines how the data
is
                                transformed.
        """
        try:
            self.glue_client.create_job(
                Name=name,
```

```
                Description=description,
                Role=role_arn,
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_job_run(self, name, input_database, input_table,
output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
tables
                               that describe the source data. This is typically
created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
the
            # Python ETL script to determine the location of input and output
data.
            response = self.glue_client.start_job_run(
                JobName=name,
                Arguments={
```

```
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRunId"]


    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobNames"]


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
        :return: The list of job runs.
        """
```

```
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRun"]


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.

        :param job_name: The name of the job definition to delete.
        """
```

```
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_database(self, name):
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
        """
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
```

```
                err.response["Error"]["Message"],
            )
            raise


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

Erstellen Sie eine Klasse, die das Szenario ausführt.

```
class GlueCrawlerJobScenario:
    """
    Encapsulates a scenario that shows how to create an AWS Glue crawler and job
 and use
    them to transform data from CSV to JSON format.
    """


    def __init__(self, glue_client, glue_service_role, glue_bucket):
        """
        :param glue_client: A Boto3 AWS Glue client.
        :param glue_service_role: An AWS Identity and Access Management (IAM)
 role
                                  that AWS Glue can assume to gain access to the
                                  resources it requires.
```

```
        :param glue_bucket: An S3 bucket that can hold a job script and output
data
                            from AWS Glue job runs.
        """
        self.glue_client = glue_client
        self.glue_service_role = glue_service_role
        self.glue_bucket = glue_bucket

    @staticmethod
    def wait(seconds, tick=12):
        """
        Waits for a specified number of seconds, while also displaying an
animated
        spinner.

        :param seconds: The number of seconds to wait.
        :param tick: The number of frames per second used to animate the spinner.
        """
        progress = "|/-\\"
        waited = 0
        while waited < seconds:
            for frame in range(tick):
                sys.stdout.write(f"\r{progress[frame % len(progress)]}")
                sys.stdout.flush()
                time.sleep(1 / tick)
            waited += 1

    def upload_job_script(self, job_script):
        """
        Uploads a Python ETL script to an S3 bucket. The script is used by the
AWS Glue
        job to transform data.

        :param job_script: The relative path to the job script.
        """
        try:
            self.glue_bucket.upload_file(Filename=job_script, Key=job_script)
            print(f"Uploaded job script '{job_script}' to the example bucket.")
        except S3UploadFailedError as err:
            logger.error("Couldn't upload job script. Here's why: %s", err)
            raise

    def run(self, crawler_name, db_name, db_prefix, data_source, job_script,
job_name):
```

```
        """
        Runs the scenario. This is an interactive experience that runs at a
command
        prompt and asks you for input throughout.

        :param crawler_name: The name of the crawler used in the scenario. If the
                             crawler does not exist, it is created.
        :param db_name: The name to give the metadata database created by the
crawler.
        :param db_prefix: The prefix to give tables added to the database by the
                          crawler.
        :param data_source: The location of the data source that is targeted by
the
                            crawler and extracted during job runs.
        :param job_script: The job script that is used to transform data during
job
                           runs.
        :param job_name: The name to give the job definition that is created
during the
                         scenario.
        """
        wrapper = GlueWrapper(self.glue_client)
        print(f"Checking for crawler {crawler_name}.")
        crawler = wrapper.get_crawler(crawler_name)
        if crawler is None:
            print(f"Creating crawler {crawler_name}.")
            wrapper.create_crawler(
                crawler_name,
                self.glue_service_role.arn,
                db_name,
                db_prefix,
                data_source,
            )
            print(f"Created crawler {crawler_name}.")
            crawler = wrapper.get_crawler(crawler_name)
        pprint(crawler)
        print("-" * 88)

        print(
            f"When you run the crawler, it crawls data stored in {data_source}
and "
            f"creates a metadata database in the AWS Glue Data Catalog that
describes "
            f"the data in the data source."
```

```
        )
        print("In this example, the source data is in CSV format.")
        ready = False
        while not ready:
            ready = Question.ask_question(
                "Ready to start the crawler? (y/n) ", Question.is_yesno
            )
        wrapper.start_crawler(crawler_name)
        print("Let's wait for the crawler to run. This typically takes a few
minutes.")
        crawler_state = None
        while crawler_state != "READY":
            self.wait(10)
            crawler = wrapper.get_crawler(crawler_name)
            crawler_state = crawler["State"]
            print(f"Crawler is {crawler['State']}.")
        print("-" * 88)

        database = wrapper.get_database(db_name)
        print(f"The crawler created database {db_name}:")
        pprint(database)
        print(f"The database contains these tables:")
        tables = wrapper.get_tables(db_name)
        for index, table in enumerate(tables):
            print(f"\t{index + 1}. {table['Name']}")
        table_index = Question.ask_question(
            f"Enter the number of a table to see more detail: ",
            Question.is_int,
            Question.in_range(1, len(tables)),
        )
        pprint(tables[table_index - 1])
        print("-" * 88)

        print(f"Creating job definition {job_name}.")
        wrapper.create_job(
            job_name,
            "Getting started example job.",
            self.glue_service_role.arn,
            f"s3://{self.glue_bucket.name}/{job_script}",
        )
        print("Created job definition.")
        print(
            f"When you run the job, it extracts data from {data_source},
transforms it "
```

```
            f"by using the {job_script} script, and loads the output into "
            f"S3 bucket {self.glue_bucket.name}."
        )
        print(
            "In this example, the data is transformed from CSV to JSON, and only
a few "
            "fields are included in the output."
        )
        job_run_status = None
        if Question.ask_question(f"Ready to run? (y/n) ", Question.is_yesno):
            job_run_id = wrapper.start_job_run(
                job_name, db_name, tables[0]["Name"], self.glue_bucket.name
            )
            print(f"Job {job_name} started. Let's wait for it to run.")
            while job_run_status not in ["SUCCEEDED", "STOPPED", "FAILED",
"TIMEOUT"]:
                self.wait(10)
                job_run = wrapper.get_job_run(job_name, job_run_id)
                job_run_status = job_run["JobRunState"]
                print(f"Job {job_name}/{job_run_id} is {job_run_status}.")
        print("-" * 88)

        if job_run_status == "SUCCEEDED":
            print(
                f"Data from your job run is stored in your S3 bucket
'{self.glue_bucket.name}':"
            )
            try:
                keys = [
                    obj.key for obj in
self.glue_bucket.objects.filter(Prefix="run-")
                ]
                for index, key in enumerate(keys):
                    print(f"\t{index + 1}: {key}")
                lines = 4
                key_index = Question.ask_question(
                    f"Enter the number of a block to download it and see the
first {lines} "
                    f"lines of JSON output in the block: ",
                    Question.is_int,
                    Question.in_range(1, len(keys)),
                )
                job_data = io.BytesIO()
                self.glue_bucket.download_fileobj(keys[key_index - 1], job_data)
```

```
                job_data.seek(0)
                for _ in range(lines):
                    print(job_data.readline().decode("utf-8"))
            except ClientError as err:
                logger.error(
                    "Couldn't get job run data. Here's why: %s: %s",
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
            print("-" * 88)

        job_names = wrapper.list_jobs()
        if job_names:
            print(f"Your account has {len(job_names)} jobs defined:")
            for index, job_name in enumerate(job_names):
                print(f"\t{index + 1}. {job_name}")
            job_index = Question.ask_question(
                f"Enter a number between 1 and {len(job_names)} to see the list
of runs for "
                f"a job: ",
                Question.is_int,
                Question.in_range(1, len(job_names)),
            )
            job_runs = wrapper.get_job_runs(job_names[job_index - 1])
            if job_runs:
                print(f"Found {len(job_runs)} runs for job {job_names[job_index -
1]}:")
                for index, job_run in enumerate(job_runs):
                    print(
                        f"\t{index + 1}. {job_run['JobRunState']} on "
                        f"{job_run['CompletedOn']:%Y-%m-%d %H:%M:%S}"
                    )
                run_index = Question.ask_question(
                    f"Enter a number between 1 and {len(job_runs)} to see details
for a run: ",
                    Question.is_int,
                    Question.in_range(1, len(job_runs)),
                )
                pprint(job_runs[run_index - 1])
            else:
                print(f"No runs found for job {job_names[job_index - 1]}")
        else:
            print("Your account doesn't have any jobs defined.")
```

```
        print("-" * 88)

        print(
            f"Let's clean up. During this example we created job definition
 '{job_name}'."
        )
        if Question.ask_question(
            "Do you want to delete the definition and all runs? (y/n) ",
            Question.is_yesno,
        ):
            wrapper.delete_job(job_name)
            print(f"Job definition '{job_name}' deleted.")
        tables = wrapper.get_tables(db_name)
        print(f"We also created database '{db_name}' that contains these
 tables:")
        for table in tables:
            print(f"\t{table['Name']}")
        if Question.ask_question(
            "Do you want to delete the tables and the database? (y/n) ",
            Question.is_yesno,
        ):
            for table in tables:
                wrapper.delete_table(db_name, table["Name"])
                print(f"Deleted table {table['Name']}.")
            wrapper.delete_database(db_name)
            print(f"Deleted database {db_name}.")
        print(f"We also created crawler '{crawler_name}'.")
        if Question.ask_question(
            "Do you want to delete the crawler? (y/n) ", Question.is_yesno
        ):
            wrapper.delete_crawler(crawler_name)
            print(f"Deleted crawler {crawler_name}.")
        print("-" * 88)


def parse_args(args):
    """
    Parse command line arguments.

    :param args: The command line arguments.
    :return: The parsed arguments.
    """
    parser = argparse.ArgumentParser(
```

```
        description="Runs the AWS Glue getting started with crawlers and jobs
 scenario. "
        "Before you run this scenario, set up scaffold resources by running "
        "'python scaffold.py deploy'."
    )
    parser.add_argument(
        "role_name",
        help="The name of an IAM role that AWS Glue can assume. This role must
 grant access "
        "to Amazon S3 and to the permissions granted by the AWSGlueServiceRole "
        "managed policy.",
    )
    parser.add_argument(
        "bucket_name",
        help="The name of an S3 bucket that AWS Glue can access to get the job
 script and "
        "put job results.",
    )
    parser.add_argument(
        "--job_script",
        default="flight_etl_job_script.py",
        help="The name of the job script file that is used in the scenario.",
    )
    return parser.parse_args(args)


def main():
    args = parse_args(sys.argv[1:])
    try:
        print("-" * 88)
        print(
            "Welcome to the AWS Glue getting started with crawlers and jobs
 scenario."
        )
        print("-" * 88)
        scenario = GlueCrawlerJobScenario(
            boto3.client("glue"),
            boto3.resource("iam").Role(args.role_name),
            boto3.resource("s3").Bucket(args.bucket_name),
        )
        scenario.upload_job_script(args.job_script)
        scenario.run(
            "doc-example-crawler",
            "doc-example-database",
```

```
            "doc-example-",
            "s3://crawler-public-us-east-1/flight/2016/csv",
            args.job_script,
            "doc-example-job",
        )
        print("-" * 88)
        print(
            "To destroy scaffold resources, including the IAM role and S3 bucket
 "
            "used in this scenario, run 'python scaffold.py destroy'."
        )
        print("\nThanks for watching!")
        print("-" * 88)
    except Exception:
        logging.exception("Something went wrong with the example.")
```

Erstellen Sie ein ETL-Skript, das AWS Glue zum Extrahieren, Transformieren und Laden von Daten während Jobausführungen verwendet wird.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
```

```
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="json",
```

```
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS -SDK für Python (Boto3).
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie eine Klasse, die die im Szenario verwendeten AWS Glue Funktionen umschließt.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
 if not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
 metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that
 the crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, _db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
      role: role_arn,
```

```
      database_name: db_name,
      targets: {
        s3_targets: [
          {
            path: s3_target
          }
        ]
      }
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create crawler: \n#{e.message}")
    raise
  end

  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
```

```
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get database #{name}: \n#{e.message}")
    raise
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: 'glueetl',
        script_location: script_location,
        python_version: '3'
      },
      glue_version: '3.0'
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
```

```
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
```

```
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
table resides.
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Removes a specified database from a Data Catalog.
  #
  # @param database_name [String] The name of the database to delete.
  # @return [void]
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end

  # Uploads a job script file to an S3 bucket.
  #
  # @param file_path [String] The local path of the job script file.
  # @param bucket_resource [Aws::S3::Bucket] The S3 bucket resource to upload the
file to.
```

```
  # @return [void]
  def upload_job_script(file_path, bucket_resource)
    File.open(file_path) do |file|
      bucket_resource.client.put_object({
                                          body: file,
                                          bucket: bucket_resource.name,
                                          key: file_path
                                        })
    end
  rescue Aws::S3::Errors::S3UploadFailedError => e
    @logger.error("S3 could not upload job script: \n#{e.message}")
    raise
  end
end
```

Erstellen Sie eine Klasse, die das Szenario ausführt.

```
class GlueCrawlerJobScenario
  def initialize(glue_client, glue_service_role, glue_bucket, logger)
    @glue_client = glue_client
    @glue_service_role = glue_service_role
    @glue_bucket = glue_bucket
    @logger = logger
  end

  def run(crawler_name, db_name, db_prefix, data_source, job_script, job_name)
    wrapper = GlueWrapper.new(@glue_client, @logger)
    setup_crawler(wrapper, crawler_name, db_name, db_prefix, data_source)
    query_database(wrapper, crawler_name, db_name)
    create_and_run_job(wrapper, job_script, job_name, db_name)
  end

  private

  def setup_crawler(wrapper, crawler_name, db_name, db_prefix, data_source)
    new_step(1, 'Create a crawler')
    crawler = wrapper.get_crawler(crawler_name)
    unless crawler
      puts "Creating crawler #{crawler_name}."
      wrapper.create_crawler(crawler_name, @glue_service_role.arn, db_name,
 db_prefix, data_source)
      puts "Successfully created #{crawler_name}."
```

```
    end
    wrapper.start_crawler(crawler_name)
    monitor_crawler(wrapper, crawler_name)
  end

  def monitor_crawler(wrapper, crawler_name)
    new_step(2, 'Monitor Crawler')
    crawler_state = nil
    until crawler_state == 'READY'
      custom_wait(15)
      crawler = wrapper.get_crawler(crawler_name)
      crawler_state = crawler[0]['state']
      print "Crawler status: #{crawler_state}".yellow
    end
  end

  def query_database(wrapper, _crawler_name, db_name)
    new_step(3, 'Query the database.')
    wrapper.get_database(db_name)
    puts "The crawler created database #{db_name}:"
    puts "Database contains tables: #{wrapper.get_tables(db_name).map { |t|
 t['name'] }}"
  end

  def create_and_run_job(wrapper, job_script, job_name, db_name)
    new_step(4, 'Create and run job.')
    wrapper.upload_job_script(job_script, @glue_bucket)
    wrapper.create_job(job_name, 'ETL Job', @glue_service_role.arn, "s3://
#{@glue_bucket.name}/#{job_script}")
    run_job(wrapper, job_name, db_name)
  end

  def run_job(wrapper, job_name, db_name)
    new_step(5, 'Run the job.')
    wrapper.start_job_run(job_name, db_name, wrapper.get_tables(db_name)[0]
['name'], @glue_bucket.name)
    job_run_status = nil
    until %w[SUCCEEDED FAILED STOPPED].include?(job_run_status)
      custom_wait(10)
      job_run = wrapper.get_job_runs(job_name)
      job_run_status = job_run[0]['job_run_state']
      print "Job #{job_name} status: #{job_run_status}".yellow
    end
  end
```

```
end

def main
  banner('../../helpers/banner.txt')
  puts 'Starting AWS Glue demo...'

  # Load resource names from YAML.
  resource_names = YAML.load_file('resource_names.yaml')

  # Setup services and resources.
  iam_role = Aws::IAM::Resource.new(region: 'us-
east-1').role(resource_names['glue_service_role'])
  s3_bucket = Aws::S3::Resource.new(region: 'us-
east-1').bucket(resource_names['glue_bucket'])

  # Instantiate scenario and run.
  scenario = GlueCrawlerJobScenario.new(Aws::Glue::Client.new(region: 'us-
east-1'), iam_role, s3_bucket, @logger)
  random_suffix = rand(10**4)
  scenario.run("crawler-#{random_suffix}", "db-#{random_suffix}", "prefix-
#{random_suffix}-", 's3://data_source',
               'job_script.py', "job-#{random_suffix}")

  puts 'Demo complete.'
end
```

Erstellen Sie ein ETL-Skript, das AWS Glue zum Extrahieren, Transformieren und Laden von Daten während Jobausführungen verwendet wird.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
```

```
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
```

```
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="json",
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- API-Details finden Sie in den folgenden Themen der AWS SDK für Ruby -API-Referenz.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Erstellen Sie einen Crawler und führen Sie ihn aus, der einen öffentlichen Amazon Simple Storage Service (Amazon S3)-Bucket crawlt und eine Metadatendatenbank generiert, die die gefundenen CSV-formatierten Daten beschreibt.

```
        let create_crawler = glue
            .create_crawler()
            .name(self.crawler())
            .database_name(self.database())
            .role(self.iam_role.expose_secret())
            .targets(
                CrawlerTargets::builder()
                    .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                    .build(),
            )
            .send()
            .await;

        match create_crawler {
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::AlreadyExistsException(_) => {
                        info!("Using existing crawler");
                        Ok(())
                    }
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
            Ok(_) => Ok(()),
        }?;
```

```
        let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

        match start_crawler {
            Ok(_) => Ok(()),
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
        }?;
```

Listen Sie Informationen zu Datenbanken und Tabellen in Ihrem auf AWS Glue Data Catalog.

```
        let database = glue
            .get_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?
            .to_owned();
        let database = database
            .database()
            .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;

        let tables = glue
            .get_tables()
            .database_name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

Erstellen und führen Sie einen Auftrag aus, der CSV-Daten aus dem Amazon-S3-Quell-Bucket extrahiert, sie durch Entfernen und Umbenennen von Feldern transformiert und die JSON-formatierte Ausgabe in einen anderen Amazon-S3-Bucket lädt.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;

        let job_run_output = glue
            .start_job_run()
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

Löscht alle Ressourcen, die von der Demo erstellt wurden.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
        }

        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zu AWS - SDK für Rust.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable

- GetCrawler
- GetDatabase
- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

Die Datei `Package.swift`

```
// swift-tools-version: 5.9
//
// The swift-tools-version declares the minimum version of Swift required to
// build this package.

import PackageDescription

let package = Package(
    name: "glue-scenario",
    // Let Xcode know the minimum Apple platforms supported.
    platforms: [
        .macOS(.v13),
        .iOS(.v15)
    ],
    dependencies: [
```

```
        // Dependencies declare other packages that this package depends on.
        .package(
            url: "https://github.com/awslabs/aws-sdk-swift",
            from: "1.0.0"),
        .package(
            url: "https://github.com/apple/swift-argument-parser.git",
            branch: "main"
        )
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module
 or a test suite.
        // Targets can depend on other targets in this package and products
        // from dependencies.
        .executableTarget(
            name: "glue-scenario",
            dependencies: [
                .product(name: "AWSGlue", package: "aws-sdk-swift"),
                .product(name: "AWSS3", package: "aws-sdk-swift"),
                .product(name: "ArgumentParser", package: "swift-argument-
parser")
            ],
            path: "Sources")

    ]
)
```

Die Swift-Codedatei,`entry.swift`.

```
// An example that shows how to use the AWS SDK for Swift to demonstrate
// creating and using crawlers and jobs using AWS Glue.
//
// 0. Upload the Python job script to Amazon S3 so it can be used when
//    calling `startJobRun()` later.
// 1. Create a crawler, pass it the IAM role and the URL of the public Amazon
//    S3 bucket that contains the source data:
//    s3://crawler-public-us-east-1/flight/2016/csv.
// 2. Start the crawler. This takes time, so after starting it, use a loop
//    that calls `getCrawler()` until the state is "READY".
// 3. Get the database created by the crawler, and the tables in the
//    database. Display them to the user.
// 4. Create a job. Pass it the IAM role and the URL to a Python ETL script
```

```
//    previously uploaded to the user's S3 bucket.
// 5. Start a job run, passing the following custom arguments. These are
//    expected by the ETL script, so must exactly match.
//    * `--input_database: <name of the database created by the crawler>`
//    * `--input_table: <name of the table created by the crawler>`
//    * `--output_bucket_url: <URL to the scaffold bucket created for the
//      user>`
// 6. Loop and get the job run until it returns one of the following states:
//    "SUCCEEDED", "STOPPED", "FAILED", or "TIMEOUT".
// 7. Output data is stored in a group of files in the user's S3 bucket.
//    Either direct the user to their location or download a file and display
//    the results inline.
// 8. List the jobs for the user's account.
// 9. Get job run details for a job run.
// 10. Delete the demo job.
// 11. Delete the database and tables created by the example.
// 12. Delete the crawler created by the example.

import ArgumentParser
import AWSS3
import Foundation
import Smithy

import AWSClientRuntime
import AWSGlue

struct ExampleCommand: ParsableCommand {
    @Option(help: "The AWS IAM role to use for AWS Glue calls.")
    var role: String

    @Option(help: "The Amazon S3 bucket to use for this example.")
    var bucket: String

    @Option(help: "The Amazon S3 URL of the data to crawl.")
    var s3url: String = "s3://crawler-public-us-east-1/flight/2016/csv"

    @Option(help: "The Python script to run as a job with AWS Glue.")
    var script: String = "./flight_etl_job_script.py"

    @Option(help: "The AWS Region to run AWS API calls in.")
    var awsRegion = "us-east-1"

    @Option(help: "A prefix string to use when naming tables.")
    var tablePrefix = "swift-glue-basics-table"
```

```
    @Option(
        help: ArgumentHelp("The level of logging for the Swift SDK to perform."),
        completion: .list([
            "critical",
            "debug",
            "error",
            "info",
            "notice",
            "trace",
            "warning"
        ])
    )
    var logLevel: String = "error"

    static var configuration = CommandConfiguration(
        commandName: "glue-scenario",
        abstract: """
        Demonstrates various features of AWS Glue.
        """,
        discussion: """
        An example showing how to use AWS Glue to create, run, and monitor
        crawlers and jobs.
        """
    )

    /// Generate and return a unique file name that begins with the specified
    /// string.
    ///
    /// - Parameters:
    ///   - prefix: Text to use at the beginning of the returned name.
    ///
    /// - Returns: A string containing a unique filename that begins with the
    ///   specified `prefix`.
    ///
    /// The returned name uses a random number between 1 million and 1 billion to
    /// provide reasonable certainty of uniqueness for the purposes of this
    /// example.
    func tempName(prefix: String) -> String {
        return "\(prefix)-\(Int.random(in: 1000000..<1000000000))"
    }

    /// Upload a file to an Amazon S3 bucket.
    ///
```

```
   /// - Parameters:
   ///   - s3Client: The S3 client to use when uploading the file.
   ///   - path: The local path of the source file to upload.
   ///   - toBucket: The name of the S3 bucket into which to upload the file.
   ///   - key: The key (name) to give the file in the S3 bucket.
   ///
   /// - Returns: `true` if the file is uploaded successfully, otherwise
`false`.
   func uploadFile(s3Client: S3Client, path: String, toBucket: String, key:
String) async -> Bool {
       do {
           let fileData: Data = try Data(contentsOf: URL(fileURLWithPath: path))
           let dataStream = ByteStream.data(fileData)
           _ = try await s3Client.putObject(
               input: PutObjectInput(
                   body: dataStream,
                   bucket: toBucket,
                   key: key
               )
           )
       } catch {
           print("*** An unexpected error occurred uploading the script to the
Amazon S3 bucket \"\(bucket)\".")
           return false
       }

       return true
   }

   /// Create a new AWS Glue crawler.
   ///
   /// - Parameters:
   ///   - glueClient: An AWS Glue client to use for the crawler.
   ///   - crawlerName: A name for the new crawler.
   ///   - iamRole: The name of an Amazon IAM role for the crawler to use.
   ///   - s3Path: The path of an Amazon S3 folder to use as a target location.
   ///   - cronSchedule: A `cron` schedule indicating when to run the crawler.
   ///   - databaseName: The name of an AWS Glue database to operate on.
   ///
   /// - Returns: `true` if the crawler is created successfully, otherwise
`false`.
   func createCrawler(glueClient: GlueClient, crawlerName: String, iamRole:
String,
```

```
                        s3Path: String, cronSchedule: String, databaseName:
String) async -> Bool {
        let s3Target = GlueClientTypes.S3Target(path: s3url)
        let targetList = GlueClientTypes.CrawlerTargets(s3Targets: [s3Target])

        do {
            _ = try await glueClient.createCrawler(
                input: CreateCrawlerInput(
                    databaseName: databaseName,
                    description: "Created by the AWS SDK for Swift Scenario
Example for AWS Glue.",
                    name: crawlerName,
                    role: iamRole,
                    schedule: cronSchedule,
                    tablePrefix: tablePrefix,
                    targets: targetList
                )
            )
        } catch _ as AlreadyExistsException {
            print("*** A crawler named \"\(crawlerName)\" already exists.")
            return false
        } catch _ as OperationTimeoutException {
            print("*** The attempt to create the AWS Glue crawler timed out.")
            return false
        } catch {
            print("*** An unexpected error occurred creating the AWS Glue
crawler: \(error.localizedDescription)")
            return false
        }

        return true
    }

    /// Delete an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler to delete.
    ///
    /// - Returns: `true` if successful, otherwise `false`.
    func deleteCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.deleteCrawler(
                input: DeleteCrawlerInput(name: name)
```

```
            )
        } catch {
            return false
        }
        return true
    }

    /// Start running an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use when starting the crawler.
    ///   - name: The name of the crawler to start running.
    ///
    /// - Returns: `true` if the crawler is started successfully, otherwise
`false`.
    func startCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.startCrawler(
                input: StartCrawlerInput(name: name)
            )
        } catch {
            print("*** An unexpected error occurred starting the crawler.")
            return false
        }

        return true
    }

    /// Get the state of the specified AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler whose state should be returned.
    ///
    /// - Returns: A `GlueClientTypes.CrawlerState` value describing the
    ///   state of the crawler.
    func getCrawlerState(glueClient: GlueClient, name: String) async ->
GlueClientTypes.CrawlerState {
        do {
            let output = try await glueClient.getCrawler(
                input: GetCrawlerInput(name: name)
            )

            // If the crawler or its state is `nil`, report that the crawler
```

```
            // is stopping. This may not be what you want for your
            // application but it works for this one!

            guard let crawler = output.crawler else {
                return GlueClientTypes.CrawlerState.stopping
            }
            guard let state = crawler.state else {
                return GlueClientTypes.CrawlerState.stopping
            }
            return state
        } catch {
            return GlueClientTypes.CrawlerState.stopping
        }
    }

    /// Wait until the specified crawler is ready to run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler to wait for.
    ///
    /// - Returns: `true` if the crawler is ready, `false` if the client is
    ///   stopping (and will therefore never be ready).
    func waitUntilCrawlerReady(glueClient: GlueClient, name: String) async ->
Bool {
        while true {
            let state = await getCrawlerState(glueClient: glueClient, name: name)

            if state == .ready {
                return true
            } else if state == .stopping {
                return false
            }

            // Wait four seconds before trying again.

            do {
                try await Task.sleep(for: .seconds(4))
            } catch {
                print("*** Error pausing the task.")
            }
        }
    }
```

```
    /// Create a new AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name to give the new job.
    ///   - role: The IAM role for the job to use when accessing AWS services.
    ///   - scriptLocation: The AWS S3 URI of the script to be run by the job.
    ///
    /// - Returns: `true` if the job is created successfully, otherwise `false`.
    func createJob(glueClient: GlueClient, name jobName: String, role: String,
                   scriptLocation: String) async -> Bool {
        let command = GlueClientTypes.JobCommand(
            name: "glueetl",
            pythonVersion: "3",
            scriptLocation: scriptLocation
        )

        do {
            _ = try await glueClient.createJob(
                input: CreateJobInput(
                    command: command,
                    description: "Created by the AWS SDK for Swift Glue basic
scenario example.",
                    glueVersion: "3.0",
                    name: jobName,
                    numberOfWorkers: 10,
                    role: role,
                    workerType: .g1x
                )
            )
        } catch {
            return false
        }
        return true
    }

    /// Return a list of the AWS Glue jobs listed on the user's account.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - maxJobs: The maximum number of jobs to return (default: 100).
    ///
    /// - Returns: An array of strings listing the names of all available AWS
    ///   Glue jobs.
```

```
func listJobs(glueClient: GlueClient, maxJobs: Int = 100) async -> [String] {
    var jobList: [String] = []
    var nextToken: String?

    repeat {
        do {
            let output = try await glueClient.listJobs(
                input: ListJobsInput(
                    maxResults: maxJobs,
                    nextToken: nextToken
                )
            )

            guard let jobs = output.jobNames else {
                return jobList
            }

            jobList = jobList + jobs
            nextToken = output.nextToken
        } catch {
            return jobList
        }
    } while (nextToken != nil)

    return jobList
}

/// Delete an AWS Glue job.
///
/// - Parameters:
///   - glueClient: The AWS Glue client to use.
///   - jobName: The name of the job to delete.
///
/// - Returns: `true` if the job is successfully deleted, otherwise `false`.
func deleteJob(glueClient: GlueClient, name jobName: String) async -> Bool {
    do {
        _ = try await glueClient.deleteJob(
            input: DeleteJobInput(jobName: jobName)
        )
    } catch {
        return false
    }
    return true
}
```

```
    /// Create an AWS Glue database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name to give the new database.
    ///   - location: The URL of the source data to use with AWS Glue.
    ///
    /// - Returns: `true` if the database is created successfully, otherwise
`false`.
    func createDatabase(glueClient: GlueClient, name databaseName: String,
location: String) async -> Bool {
        let databaseInput = GlueClientTypes.DatabaseInput(
            description: "Created by the AWS SDK for Swift Glue basic scenario
example.",
            locationUri: location,
            name: databaseName
        )

        do {
            _ = try await glueClient.createDatabase(
                input: CreateDatabaseInput(
                    databaseInput: databaseInput
                )
            )
        } catch {
            return false
        }

        return true
    }

    /// Get the AWS Glue database with the specified name.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the database to return.
    ///
    /// - Returns: The `GlueClientTypes.Database` object describing the
    ///   specified database, or `nil` if an error occurs or the database
    ///   isn't found.
    func getDatabase(glueClient: GlueClient, name: String) async ->
GlueClientTypes.Database? {
        do {
```

```
            let output = try await glueClient.getDatabase(
                input: GetDatabaseInput(name: name)
            )

            return output.database
        } catch {
            return nil
        }
    }

    /// Returns a list of the tables in the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database whose tables are to be
    ///     returned.
    ///
    /// - Returns: An array of `GlueClientTypes.Table` objects, each
    ///   describing one table in the named database. An empty array indicates
    ///   that there are either no tables in the database, or an error
    ///   occurred before any tables could be found.
    func getTablesInDatabase(glueClient: GlueClient, databaseName: String) async
-> [GlueClientTypes.Table] {
        var tables: [GlueClientTypes.Table] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.getTables(
                    input: GetTablesInput(
                        databaseName: databaseName,
                        nextToken: nextToken
                    )
                )

                guard let tableList = output.tableList else {
                    return tables
                }

                tables = tables + tableList
                nextToken = output.nextToken
            } catch {
                return tables
            }
```

```
        } while nextToken != nil

        return tables
    }

    /// Delete the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database to delete.
    ///   - deleteTables: A Bool indicating whether or not to delete the
    ///     tables in the database before attempting to delete the database.
    ///
    /// - Returns: `true` if the database (and optionally its tables) are
    ///   deleted, otherwise `false`.
    func deleteDatabase(glueClient: GlueClient, name databaseName: String,
                        withTables deleteTables: Bool = false) async -> Bool {
        if deleteTables {
            var tableNames: [String] = []

            // Get a list of the names of all of the tables in the database.

            let tableList = await self.getTablesInDatabase(glueClient:
glueClient, databaseName: databaseName)
            for table in tableList {
                guard let name = table.name else {
                    continue
                }
                tableNames.append(name)
            }

            // Delete the tables. If there's only one table, use
            // `deleteTable()`, otherwise, use `batchDeleteTable()`. You can
            // use `batchDeleteTable()` for a single table, but this
            // demonstrates the use of `deleteTable()`.

            if tableNames.count == 1 {
                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
                            name: tableNames[0]
                        )
```

```
                    )
                } catch {
                    print("*** Unable to delete the table.")
                }
            } else {
                do {
                    print("    Deleting tables...")
                    _ = try await glueClient.batchDeleteTable(
                        input: BatchDeleteTableInput(
                            databaseName: databaseName,
                            tablesToDelete: tableNames
                        )
                    )
                } catch {
                    print("*** Unable to delete the tables.")
                }
            }
        }

        // Delete the database itself.

        do {
            print("    Deleting the database itself...")
            _ = try await glueClient.deleteDatabase(
                input: DeleteDatabaseInput(name: databaseName)
            )
        } catch {
            print("*** Unable to delete the database.")
            return false
        }
        return true
    }

    /// Start an AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to run.
    ///   - databaseName: The name of the AWS Glue database to run the job
against.
    ///   - tableName: The name of the table in the database to run the job
against.
    ///   - outputURL: The AWS S3 URI of the bucket location into which to
    ///     write the resulting output.
```

```
    ///
    /// - Returns: `true` if the job run is started successfully, otherwise
`false`.
    func startJobRun(glueClient: GlueClient, name jobName: String, databaseName:
String,
                     tableName: String, outputURL: String) async -> String? {
        do {
            let output = try await glueClient.startJobRun(
                input: StartJobRunInput(
                    arguments: [
                        "--input_database": databaseName,
                        "--input_table": tableName,
                        "--output_bucket_url": outputURL
                    ],
                    jobName: jobName,
                    numberOfWorkers: 10,
                    workerType: .g1x
                )
            )

            guard let id = output.jobRunId else {
                return nil
            }

            return id
        } catch {
            return nil
        }
    }

    /// Return a list of the job runs for the specified job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job for which to return its job runs.
    ///   - maxResults: The maximum number of job runs to return (default:
    ///     1000).
    ///
    /// - Returns: An array of `GlueClientTypes.JobRun` objects describing
    ///   each job run.
    func getJobRuns(glueClient: GlueClient, name jobName: String, maxResults:
Int? = nil) async -> [GlueClientTypes.JobRun] {
        do {
            let output = try await glueClient.getJobRuns(
```

```
                input: GetJobRunsInput(
                    jobName: jobName,
                    maxResults: maxResults
                )
            )

            guard let jobRuns = output.jobRuns else {
                print("*** No job runs found.")
                return []
            }

            return jobRuns
        } catch is EntityNotFoundException {
            print("*** The specified job name, \(jobName), doesn't exist.")
            return []
        } catch {
            print("*** Unexpected error getting job runs:")
            dump(error)
            return []
        }
    }

    /// Get information about a specific AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to return job run data for.
    ///   - id: The run ID of the specific job run to return.
    ///
    /// - Returns: A `GlueClientTypes.JobRun` object describing the state of
    ///   the job run, or `nil` if an error occurs.
    func getJobRun(glueClient: GlueClient, name jobName: String, id: String)
async -> GlueClientTypes.JobRun? {
        do {
            let output = try await glueClient.getJobRun(
                input: GetJobRunInput(
                    jobName: jobName,
                    runId: id
                )
            )

            return output.jobRun
        } catch {
            return nil
```

```
        }
    }

    /// Called by ``main()`` to run the bulk of the example.
    func runAsync() async throws {
        // A name to give the Python script upon upload to the Amazon S3
        // bucket.
        let scriptName = "jobscript.py"

        // Schedule string in `cron` format, as described here:
        // https://docs.aws.amazon.com/glue/latest/dg/monitor-data-warehouse-
schedule.html
        let cron = "cron(15 12 * * ? *)"

        let glueConfig = try await GlueClient.GlueClientConfiguration(region:
 awsRegion)
        let glueClient = GlueClient(config: glueConfig)

        let s3Config = try await S3Client.S3ClientConfiguration(region:
 awsRegion)
        let s3Client = S3Client(config: s3Config)

        // Create random names for things that need them.

        let crawlerName = tempName(prefix: "swift-glue-basics-crawler")
        let databaseName = tempName(prefix: "swift-glue-basics-db")

        // Create a name for the AWS Glue job.

        let jobName = tempName(prefix: "scenario-job")

        // The URL of the Python script on S3.

        let scriptURL = "s3://\(bucket)/\(scriptName)"

        print("Welcome to the AWS SDK for Swift basic scenario for AWS Glue!")

        //===========================================================================
        // 0. Upload the Python script to the target bucket so it's available
        //    for use by the Amazon Glue service.
        //===========================================================================

        print("Uploading the Python script: \(script) as key \(scriptName)")
        print("Destination bucket: \(bucket)")
```

```
        if !(await uploadFile(s3Client: s3Client, path: script, toBucket: bucket,
key: scriptName)) {
            return
        }

        //======================================================================
        // 1. Create the database and crawler using the randomized names
        //    generated previously.
        //======================================================================

        print("Creating database \"\(databaseName)\"...")
        if !(await createDatabase(glueClient: glueClient, name: databaseName,
location: s3url)) {
            print("*** Unable to create the database.")
            return
        }

        print("Creating crawler \"\(crawlerName)\"...")
        if !(await createCrawler(glueClient: glueClient, crawlerName:
crawlerName,
                                 iamRole: role, s3Path: s3url, cronSchedule:
cron,
                                 databaseName: databaseName)) {
            return
        }

        //======================================================================
        // 2. Start the crawler, then wait for it to be ready.
        //======================================================================

        print("Starting the crawler and waiting until it's ready...")
        if !(await startCrawler(glueClient: glueClient, name: crawlerName)) {
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        if !(await waitUntilCrawlerReady(glueClient: glueClient, name:
crawlerName)) {
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
        }

        //======================================================================
        // 3. Get the database and table created by the crawler.
        //======================================================================
```

```
        print("Getting the crawler's database...")
        let database = await getDatabase(glueClient: glueClient, name:
databaseName)

        guard let database else {
            print("*** Unable to get the database.")
            return
        }
        print("Database URI: \(database.locationUri ?? "<unknown>")")

        let tableList = await getTablesInDatabase(glueClient: glueClient,
databaseName: databaseName)

        print("Found \(tableList.count) table(s):")
        for table in tableList {
            print("  \(table.name ?? "<unnamed>")")
        }

        if tableList.count != 1 {
            print("*** Incorrect number of tables found. There should only be
one.")
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        guard let tableName = tableList[0].name else {
            print("*** Table is unnamed.")
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        //====================================================================
        // 4. Create a job.
        //====================================================================

        print("Creating a job...")
        if !(await createJob(glueClient: glueClient, name: jobName, role: role,
                             scriptLocation: scriptURL)) {
```

```
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        //====================================================================
        // 5. Start a job run.
        //====================================================================

        print("Starting the job...")

        // Construct the Amazon S3 URL for the job run's output. This is in
        // the bucket specified on the command line, with a folder name that's
        // unique for this job run.

        let timeStamp = Date().timeIntervalSince1970
        let jobPath = "\(jobName)-\(Int(timeStamp))"
        let outputURL = "s3://\(bucket)/\(jobPath)"

        // Start the job run.

        let jobRunID = await startJobRun(glueClient: glueClient, name: jobName,
                                         databaseName: databaseName,
                                         tableName: tableName,
                                         outputURL: outputURL)

        guard let jobRunID else {
            print("*** Job run ID is invalid.")
            _ = await deleteJob(glueClient: glueClient, name: jobName)
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        //====================================================================
        // 6. Wait for the job run to indicate that the run is complete.
        //====================================================================

        print("Waiting for job run to end...")

        var jobRunFinished = false
        var jobRunState: GlueClientTypes.JobRunState
```

```
        repeat {
            let jobRun = await getJobRun(glueClient: glueClient, name: jobName,
 id: jobRunID)
            guard let jobRun else {
                print("*** Unable to get the job run.")
                _ = await deleteJob(glueClient: glueClient, name: jobName)
                _ = await deleteDatabase(glueClient: glueClient, name:
 databaseName, withTables: true)
                _ = await deleteCrawler(glueClient: glueClient, name:
 crawlerName)
                return
            }
            jobRunState = jobRun.jobRunState ?? .failed

            //
=====================================================================
            // 7. Output where to find the data if the job run was successful.
            //    If the job run failed for any reason, output an appropriate
            //    error message.
            //
=====================================================================

            switch jobRunState {
                case .succeeded:
                    print("Job run succeeded. JSON files are in the Amazon S3
 path:")
                    print("    \(outputURL)")
                    jobRunFinished = true
                case .stopped:
                    jobRunFinished = true
                case .error:
                    print("*** Error: Job run ended in an error.
 \(jobRun.errorMessage ?? "")")
                    jobRunFinished = true
                case .failed:
                    print("*** Error: Job run failed. \(jobRun.errorMessage ??
 "")")
                    jobRunFinished = true
                case .timeout:
                    print("*** Warning: Job run timed out.")
                    jobRunFinished = true
                default:
                    do {
```

```
                    try await Task.sleep(for: .milliseconds(250))
                } catch {
                    print("*** Error pausing the task.")
                }
            }
        } while jobRunFinished != true

        //=========================================================================
        // 7.5. List the job runs for this job, showing each job run's ID and
        // its execution time.
        //=========================================================================

        print("Getting all job runs for the job \(jobName):")
        let jobRuns = await getJobRuns(glueClient: glueClient, name: jobName)

        if jobRuns.count == 0 {
            print("    <no job runs found>")
        } else {
            print("Found \(jobRuns.count) job runs... listing execution times:")
            for jobRun in jobRuns {
                print("    \(jobRun.id ?? "<unnamed>"): \(jobRun.executionTime)
seconds")
            }
        }

        //=========================================================================
        // 8. List the jobs for the user's account.
        //=========================================================================

        print("\nThe account has the following jobs:")
        let jobs = await listJobs(glueClient: glueClient)

        if jobs.count == 0 {
            print("    <no jobs found>")
        } else {
            for job in jobs {
                print("    \(job)")
            }
        }

        //=========================================================================
        // 9. Get the job run details for a job run.
        //=========================================================================
```

```
        print("Information about the job run:")
        let jobRun = await getJobRun(glueClient: glueClient, name: jobName, id:
jobRunID)

        guard let jobRun else {
            print("*** Unable to retrieve the job run.")
            _ = await deleteJob(glueClient: glueClient, name: jobName)
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        let startDate = jobRun.startedOn ?? Date(timeIntervalSince1970: 0)
        let endDate = jobRun.completedOn ?? Date(timeIntervalSince1970: 0)
        let dateFormatter: DateFormatter = DateFormatter()
        dateFormatter.dateStyle = .long
        dateFormatter.timeStyle = .long

        print("    Started at: \(dateFormatter.string(from: startDate))")
        print("  Completed at: \(dateFormatter.string(from: endDate))")

        //===========================================================================
        // 10. Delete the job.
        //===========================================================================

        print("\nDeleting the job...")
        _ = await deleteJob(glueClient: glueClient, name: jobName)

        //===========================================================================
        // 11. Delete the database and tables created by this example.
        //===========================================================================

        print("Deleting the database...")
        _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)

        //===========================================================================
        // 12. Delete the crawler.
        //===========================================================================

        print("Deleting the crawler...")
        if !(await deleteCrawler(glueClient: glueClient, name: crawlerName)) {
            return
```

```
        }
    }
}

/// The program's asynchronous entry point.
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Weitere API-Informationen finden Sie in den folgenden Themen der API-Referenz zum AWS SDK für Swift.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler

- StartJobRun

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Aktionen zur AWS Glue Verwendung AWS SDKs

Die folgenden Codebeispiele zeigen, wie Sie einzelne AWS Glue Aktionen mit ausführen AWS SDKs. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der AWS Glue -API-Referenz.

Beispiele

- Verwendung CreateCrawler mit einem AWS SDK
- Verwendung CreateJob mit einem AWS SDK oder CLI
- Verwendung DeleteCrawler mit einem AWS SDK
- Verwendung DeleteDatabase mit einem AWS SDK
- Verwendung DeleteJob mit einem AWS SDK oder CLI
- Verwendung DeleteTable mit einem AWS SDK
- Verwendung GetCrawler mit einem AWS SDK
- Verwendung GetDatabase mit einem AWS SDK
- Verwendung GetDatabases mit einem AWS SDK oder CLI
- Verwendung GetJob mit einem AWS SDK oder CLI
- Verwendung GetJobRun mit einem AWS SDK oder CLI
- Verwendung GetJobRuns mit einem AWS SDK oder CLI
- Verwendung GetTables mit einem AWS SDK oder CLI
- Verwendung ListJobs mit einem AWS SDK
- Verwendung StartCrawler mit einem AWS SDK oder CLI
- Verwendung StartJobRun mit einem AWS SDK oder CLI

Verwendung **`CreateCrawler`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `CreateCrawler` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /// <summary>
   /// Create an AWS Glue crawler.
   /// </summary>
   /// <param name="crawlerName">The name for the crawler.</param>
   /// <param name="crawlerDescription">A description of the crawler.</param>
   /// <param name="role">The AWS Identity and Access Management (IAM) role to
   /// be assumed by the crawler.</param>
   /// <param name="schedule">The schedule on which the crawler will be
executed.</param>
   /// <param name="s3Path">The path to the Amazon Simple Storage Service
(Amazon S3)
   /// bucket where the Python script has been stored.</param>
   /// <param name="dbName">The name to use for the database that will be
   /// created by the crawler.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> CreateCrawlerAsync(
       string crawlerName,
       string crawlerDescription,
       string role,
       string schedule,
       string s3Path,
       string dbName)
   {
```

```
        var s3Target = new S3Target
        {
            Path = s3Path,
        };

        var targetList = new List<S3Target>
        {
            s3Target,
        };

        var targets = new CrawlerTargets
        {
            S3Targets = targetList,
        };

        var crawlerRequest = new CreateCrawlerRequest
        {
            DatabaseName = dbName,
            Name = crawlerName,
            Description = crawlerDescription,
            Targets = targets,
            Role = role,
            Schedule = schedule,
        };

        var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Creates a new AWS Glue crawler using the AWS Glue Java API.
    *
    * @param glueClient  the AWS Glue client used to interact with the AWS Glue
service
    * @param iam         the IAM role that the crawler will use to access the
data source
    * @param s3Path      the S3 path that the crawler will scan for data
    * @param cron        the cron expression that defines the crawler's schedule
    * @param dbName      the name of the AWS Glue database where the crawler
will store the metadata
    * @param crawlerName the name of the crawler to be created
    */
   public static void createGlueCrawler(GlueClient glueClient,
                                        String iam,
                                        String s3Path,
                                        String cron,
                                        String dbName,
                                        String crawlerName) {

       try {
           S3Target s3Target = S3Target.builder()
               .path(s3Path)
               .build();

           List<S3Target> targetList = new ArrayList<>();
           targetList.add(s3Target);
           CrawlerTargets targets = CrawlerTargets.builder()
               .s3Targets(targetList)
               .build();
```

```
            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                .databaseName(dbName)
                .name(crawlerName)
                .description("Created by the AWS Glue Java API")
                .targets(targets)
                .role(iam)
                .schedule(cron)
                .build();

            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
            throw e;
        }
    }
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});

  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });
```

```
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun createGlueCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    // Add the S3Target to a list.
    val targetList = mutableListOf<S3Target>()
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val request =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
```

```
            description = "Created by the AWS Glue Kotlin API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient.fromEnvironment { region = "us-west-2" }.use { glueClient ->
        glueClient.createCrawler(request)
        println("$crawlerName was successfully created")
    }
}
```

- Einzelheiten zur API finden Sie CreateCrawlerin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);

   public function createCrawler($crawlerName, $role, $databaseName, $path):
Result
   {
        return $this->customWaiter(function () use ($crawlerName, $role,
$databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
```

```
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
        });
    }
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
 data
        in the target.

        :param name: The name of the crawler.
```

```
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
Access
                         Management (IAM) role that grants permission to let AWS
Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
crawler.
        :param db_prefix: The prefix to give any database tables that are created
by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
                          the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie CreateCrawlerin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
 metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that
 the crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, _db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
      role: role_arn,
      database_name: db_name,
      targets: {
        s3_targets: [
          {
            path: s3_target
```

```
        }
      ]
    }
  )
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not create crawler: \n#{e.message}")
  raise
end
```

- Einzelheiten zur API finden Sie CreateCrawlerin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    let create_crawler = glue
        .create_crawler()
        .name(self.crawler())
        .database_name(self.database())
        .role(self.iam_role.expose_secret())
        .targets(
            CrawlerTargets::builder()
                .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                .build(),
        )
        .send()
        .await;

    match create_crawler {
        Err(err) => {
            let glue_err: aws_sdk_glue::Error = err.into();
            match glue_err {
                aws_sdk_glue::Error::AlreadyExistsException(_) => {
                    info!("Using existing crawler");
                    Ok(())
```

```
                    }
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
            Ok(_) => Ok(()),
        }?;
```

- Einzelheiten zur API finden Sie CreateCrawlerin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Create a new AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: An AWS Glue client to use for the crawler.
    ///   - crawlerName: A name for the new crawler.
    ///   - iamRole: The name of an Amazon IAM role for the crawler to use.
    ///   - s3Path: The path of an Amazon S3 folder to use as a target location.
    ///   - cronSchedule: A `cron` schedule indicating when to run the crawler.
    ///   - databaseName: The name of an AWS Glue database to operate on.
    ///
    /// - Returns: `true` if the crawler is created successfully, otherwise
 `false`.
    func createCrawler(glueClient: GlueClient, crawlerName: String, iamRole:
 String,
                       s3Path: String, cronSchedule: String, databaseName:
 String) async -> Bool {
        let s3Target = GlueClientTypes.S3Target(path: s3url)
        let targetList = GlueClientTypes.CrawlerTargets(s3Targets: [s3Target])
```

```
        do {
            _ = try await glueClient.createCrawler(
                input: CreateCrawlerInput(
                    databaseName: databaseName,
                    description: "Created by the AWS SDK for Swift Scenario
Example for AWS Glue.",
                    name: crawlerName,
                    role: iamRole,
                    schedule: cronSchedule,
                    tablePrefix: tablePrefix,
                    targets: targetList
                )
            )
        } catch _ as AlreadyExistsException {
            print("*** A crawler named \"\(crawlerName)\" already exists.")
            return false
        } catch _ as OperationTimeoutException {
            print("*** The attempt to create the AWS Glue crawler timed out.")
            return false
        } catch {
            print("*** An unexpected error occurred creating the AWS Glue
crawler: \(error.localizedDescription)")
            return false
        }

        return true
    }
```

- Einzelheiten zur API finden Sie CreateCrawlerin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`CreateJob`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `CreateJob` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Create an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="roleName">The name of the IAM role to be assumed by
    /// the job.</param>
    /// <param name="description">A description of the job.</param>
    /// <param name="scriptUrl">The URL to the script.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
    {
        var command = new JobCommand
        {
            PythonVersion = "3",
            Name = "glueetl",
            ScriptLocation = scriptUrl,
        };

        var arguments = new Dictionary<string, string>
        {
            { "--input_database", dbName },
            { "--input_table", tableName },
            { "--output_bucket_url", bucketUrl }
        };

        var request = new CreateJobRequest
        {
            Command = command,
```

```
            DefaultArguments = arguments,
            Description = description,
            GlueVersion = "3.0",
            Name = jobName,
            NumberOfWorkers = 10,
            Role = roleName,
            WorkerType = "G.1X"
        };

        var response = await _amazonGlue.CreateJobAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::CreateJobRequest request;
       request.SetName(JOB_NAME);
       request.SetRole(roleArn);
       request.SetGlueVersion(GLUE_VERSION);

       Aws::Glue::Model::JobCommand command;
       command.SetName(JOB_COMMAND_NAME);
       command.SetPythonVersion(JOB_PYTHON_VERSION);
```

```
        command.SetScriptLocation(
                Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
        request.SetCommand(command);

        Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the job." << std::endl;
        }
        else {
            std::cerr << "Error creating the job. " <<
 outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Einen Auftrag zur Datentransformation erstellen

Im folgenden Beispiel für `create-job` wird ein Streaming-Job erstellt, der ein in S3 gespeichertes Skript ausführt.

```
aws glue create-job \
    --name my-testing-job \
    --role AWSGlueServiceRoleDefault \
    --command '{ \
        "Name": "gluestreaming", \
        "ScriptLocation": "s3://amzn-s3-demo-bucket/folder/" \
    }' \
    --region us-east-1 \
    --output json \
    --default-arguments '{ \
        "--job-language":"scala", \
        "--class":"GlueApp" \
    }' \
```

```
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

Inhalt von `test_script.scala`:

```
import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
    def main(sysArgs: Array[String]) {
        val spark: SparkContext = new SparkContext()
        val glueContext: GlueContext = new GlueContext(spark)
        // @params: [JOB_NAME]
        val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME").toArray)
        Job.init(args("JOB_NAME"), glueContext, args.asJava)
        // @type: DataSource
        // @args: [database = "tempdb", table_name = "s3-source",
 transformation_ctx = "datasource0"]
        // @return: datasource0
        // @inputs: []
        val datasource0 = glueContext.getCatalogSource(database = "tempdb",
 tableName = "s3-source", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
        // @type: ApplyMapping
        // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
 ("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
 "status", "string")], transformation_ctx = "applymapping1"]
        // @return: applymapping1
        // @inputs: [frame = datasource0]
        val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
 "int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
 "int"), ("status", "string", "status", "string")), caseSensitive = false,
 transformationContext = "applymapping1")
        // @type: SelectFields
```

```
        // @args: [paths = ["sensorid", "currenttemperature", "status"],
 transformation_ctx = "selectfields2"]
        // @return: selectfields2
        // @inputs: [frame = applymapping1]
        val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
 "currenttemperature", "status"), transformationContext = "selectfields2")
        // @type: ResolveChoice
        // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name =
 "my-s3-sink", transformation_ctx = "resolvechoice3"]
        // @return: resolvechoice3
        // @inputs: [frame = selectfields2]
        val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
 Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
 Some("my-s3-sink"), transformationContext = "resolvechoice3")
        // @type: DataSink
        // @args: [database = "tempdb", table_name = "my-s3-sink",
 transformation_ctx = "datasink4"]
        // @return: datasink4
        // @inputs: [frame = resolvechoice3]
        val datasink4 = glueContext.getCatalogSink(database = "tempdb",
 tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
 "datasink4").writeDynamicFrame(resolvechoice3)
        Job.commit()
    }
}
```

Ausgabe:

```
{
    "Name": "my-testing-job"
}
```

Weitere Informationen finden Sie unter Authoring Jobs in AWS Glue im AWS Glue Developer Guide.

- Einzelheiten zur API finden Sie CreateJobin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Creates a new AWS Glue job.
    *
    * @param glueClient     the AWS Glue client to use for the operation
    * @param jobName        the name of the job to create
    * @param iam            the IAM role to associate with the job
    * @param scriptLocation the location of the script to be used by the job
    * @throws GlueException if there is an error creating the job
    */
   public static void createJob(GlueClient glueClient, String jobName, String
iam, String scriptLocation) {
       try {
           JobCommand command = JobCommand.builder()
               .pythonVersion("3")
               .name("glueetl")
               .scriptLocation(scriptLocation)
               .build();

           CreateJobRequest jobRequest = CreateJobRequest.builder()
               .description("A Job created by using the AWS SDK for Java V2")
               .glueVersion("2.0")
               .workerType(WorkerType.G_1_X)
               .numberOfWorkers(10)
               .name(jobName)
               .role(iam)
               .command(command)
               .build();

           glueClient.createJob(jobRequest);
           System.out.println(jobName + " was successfully created.");
```

```
        } catch (GlueException e) {
            throw e;
        }
    }
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $jobName = 'test-job-' . $uniqid;

        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
$scriptLocation);

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
'3', $glueVersion = '3.0'): Result
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK für PHP API-Referenz.

PowerShell

Tools für PowerShell V4

Beispiel 1: Dieses Beispiel erstellt einen neuen Job in AWS Glue. Der Wert des Befehlsnamens ist immer**`glueetl`**. AWS Glue unterstützt das Ausführen von Jobskripten, die in Python oder Scala geschrieben wurden. In diesem Beispiel ist das Jobskript (MyTestGlueJob.py) in Python geschrieben. Python-Parameter werden in der **`$DefArgs`** Variablen angegeben und dann an den PowerShell Befehl im **`DefaultArguments`** Parameter übergeben, der eine Hashtabelle akzeptiert. Die Parameter in der **`$JobParams`** Variablen stammen aus der CreateJob API, die im Thema Jobs (https://docs.aws.amazon.com/glue/latest/dg/aws- glue-api-jobs-job .html) der AWS Glue-API-Referenz dokumentiert ist.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/
MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
     '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
     '--job-bookmark-option' = 'job-bookmark-disable'
     '--job-language' = 'python'
     }
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"  = $DefArgs
    "Description"       = "This is a test"
    "ExecutionProperty" = $ExecutionProp
    "MaxRetries"        = "1"
```

```
    "Name"               = "MyOregonTestGlueJob"
    "Role"               = "Amazon-GlueServiceRoleForSSM"
    "Timeout"            = "20"
     }

New-GlueJob @JobParams
```

- Einzelheiten zur API finden Sie unter CreateJob AWS -Tools für PowerShellCmdlet-Referenz (V4).

Tools für V5 PowerShell

Beispiel 1: Dieses Beispiel erstellt einen neuen Job in AWS Glue. Der Wert des Befehlsnamens ist immer**`glueetl`**. AWS Glue unterstützt das Ausführen von Jobskripten, die in Python oder Scala geschrieben wurden. In diesem Beispiel ist das Jobskript (MyTestGlueJob.py) in Python geschrieben. Python-Parameter werden in der **`$DefArgs`** Variablen angegeben und dann an den PowerShell Befehl im **`DefaultArguments`** Parameter übergeben, der eine Hashtabelle akzeptiert. Die Parameter in der **`$JobParams`** Variablen stammen aus der CreateJob API, die im Thema Jobs (https://docs.aws.amazon.com/glue/latest/dg/aws- glue-api-jobs-job .html) der AWS Glue-API-Referenz dokumentiert ist.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/
MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
     '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
     '--job-bookmark-option' = 'job-bookmark-disable'
     '--job-language' = 'python'
     }
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
```

```
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"  = $DefArgs
    "Description"       = "This is a test"
    "ExecutionProperty" = $ExecutionProp
    "MaxRetries"        = "1"
    "Name"              = "MyOregonTestGlueJob"
    "Role"              = "Amazon-GlueServiceRoleForSSM"
    "Timeout"           = "20"
     }

New-GlueJob @JobParams
```

- Einzelheiten zur API finden Sie unter CreateJob AWS -Tools für PowerShellCmdlet-Referenz (V5).

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
 that can
```

```
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
run as
                                part of the job. The script defines how the data
is
                                transformed.
        """
        try:
            self.glue_client.create_job(
                Name=name,
                Description=description,
                Role=role_arn,
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie CreateJobin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: 'glueetl',
        script_location: script_location,
        python_version: '3'
      },
      glue_version: '3.0'
    )
```

```
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie CreateJobin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;
```

- Einzelheiten zur API finden Sie CreateJobin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Create a new AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name to give the new job.
    ///   - role: The IAM role for the job to use when accessing AWS services.
    ///   - scriptLocation: The AWS S3 URI of the script to be run by the job.
    ///
    /// - Returns: `true` if the job is created successfully, otherwise `false`.
    func createJob(glueClient: GlueClient, name jobName: String, role: String,
                   scriptLocation: String) async -> Bool {
        let command = GlueClientTypes.JobCommand(
            name: "glueetl",
            pythonVersion: "3",
            scriptLocation: scriptLocation
        )

        do {
            _ = try await glueClient.createJob(
                input: CreateJobInput(
                    command: command,
                    description: "Created by the AWS SDK for Swift Glue basic
 scenario example.",
                    glueVersion: "3.0",
                    name: jobName,
                    numberOfWorkers: 10,
                    role: role,
                    workerType: .g1x
                )
```

```
            )
        } catch {
            return false
        }
        return true
    }
```

- Einzelheiten zur API finden Sie CreateJobin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteCrawler`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `DeleteCrawler` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Delete an AWS Glue crawler.
/// </summary>
/// <param name="crawlerName">The name of the crawler.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DeleteCrawlerAsync(string crawlerName)
{
```

```
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Deletes a specific AWS Glue crawler.
    *
    * @param glueClient  the AWS Glue client object
    * @param crawlerName the name of the crawler to be deleted
    * @throws GlueException if an error occurs during the deletion process
    */
   public static void deleteSpecificCrawler(GlueClient glueClient, String
crawlerName) {
       try {
           DeleteCrawlerRequest deleteCrawlerRequest =
DeleteCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           glueClient.deleteCrawler(deleteCrawlerRequest);
           System.out.println(crawlerName + " was deleted");

       } catch (GlueException e) {
           throw e;
       }
   }
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

    public function deleteCrawler($crawlerName)
    {
```

```
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
```

```
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteCrawlerin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end
```

```
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Delete an AWS Glue crawler.
    ///
```

```
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler to delete.
    ///
    /// - Returns: `true` if successful, otherwise `false`.
    func deleteCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.deleteCrawler(
                input: DeleteCrawlerInput(name: name)
            )
        } catch {
            return false
        }
        return true
    }
```

- Einzelheiten zur API finden Sie DeleteCrawlerin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteDatabase`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `DeleteDatabase` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Delete the AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteDatabaseRequest request;
        request.SetName(database);

        Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
                request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the database." << std::endl;
```

```
        }
        else {
            std::cerr << "Error deleting database. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /**
     * Deletes a AWS Glue Database.
     *
     * @param glueClient   An instance of the AWS Glue client used to interact
with the AWS Glue service.
     * @param databaseName The name of the database to be deleted.
     * @throws GlueException If an error occurs while deleting the database.
     */
    public static void deleteDatabase(GlueClient glueClient, String databaseName)
{
        try {
            DeleteDatabaseRequest request = DeleteDatabaseRequest.builder()
                .name(databaseName)
                .build();

            glueClient.deleteDatabase(request);
            System.out.println(databaseName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
```

```
    }
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_database(self, name):
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
        """
```

```
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteDatabasein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Removes a specified database from a Data Catalog.
  #
```

```
  # @param database_name [String] The name of the database to delete.
  # @return [void]
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end
```

- Einzelheiten zur API finden Sie DeleteDatabasein der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Einzelheiten zur API finden Sie DeleteDatabasein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Delete the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database to delete.
    ///   - deleteTables: A Bool indicating whether or not to delete the
    ///     tables in the database before attempting to delete the database.
    ///
    /// - Returns: `true` if the database (and optionally its tables) are
    ///   deleted, otherwise `false`.
    func deleteDatabase(glueClient: GlueClient, name databaseName: String,
                        withTables deleteTables: Bool = false) async -> Bool {
        if deleteTables {
            var tableNames: [String] = []

            // Get a list of the names of all of the tables in the database.

            let tableList = await self.getTablesInDatabase(glueClient:
 glueClient, databaseName: databaseName)
            for table in tableList {
                guard let name = table.name else {
                    continue
                }
                tableNames.append(name)
            }

            // Delete the tables. If there's only one table, use
            // `deleteTable()`, otherwise, use `batchDeleteTable()`. You can
            // use `batchDeleteTable()` for a single table, but this
            // demonstrates the use of `deleteTable()`.

            if tableNames.count == 1 {
                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
                            name: tableNames[0]
                        )
                    )
```

```
                } catch {
                    print("*** Unable to delete the table.")
                }
            } else {
                do {
                    print("    Deleting tables...")
                    _ = try await glueClient.batchDeleteTable(
                        input: BatchDeleteTableInput(
                            databaseName: databaseName,
                            tablesToDelete: tableNames
                        )
                    )
                } catch {
                    print("*** Unable to delete the tables.")
                }
            }
        }

        // Delete the database itself.

        do {
            print("    Deleting the database itself...")
            _ = try await glueClient.deleteDatabase(
                input: DeleteDatabaseInput(name: databaseName)
            )
        } catch {
            print("*** Unable to delete the database.")
            return false
        }
        return true
    }
```

- Einzelheiten zur API finden Sie DeleteDatabasein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteJob`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteJob` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteJobRequest request;
        request.SetJobName(job);

        Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the job." << std::endl;
        }
        else {
            std::cerr << "Error deleting the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Einen Auftrag löschen

Das folgende Beispiel für `delete-job` löscht einen Auftrag, der nicht mehr benötigt wird.

```
aws glue delete-job \
    --job-name my-testing-job
```

Ausgabe:

```
{
    "JobName": "my-testing-job"
}
```

Weitere Informationen finden Sie unter Arbeiten mit Jobs auf der AWS Glue-Konsole im AWS Glue-Entwicklerhandbuch.

- Einzelheiten zur API finden Sie DeleteJobin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
 * Deletes a Glue job.
 *
 * @param glueClient the Glue client to use for the operation
 * @param jobName    the name of the job to be deleted
 * @throws GlueException if there is an error deleting the job
 */
public static void deleteJob(GlueClient glueClient, String jobName) {
    try {
        DeleteJobRequest jobRequest = DeleteJobRequest.builder()
            .jobName(jobName)
            .build();
```

```
            glueClient.deleteJob(jobRequest);
            System.out.println(jobName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
    }
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        echo "Delete the job.\n";
        $glueClient->deleteJob([
            'JobName' => $job['Name'],
        ]);

    public function deleteJob($jobName)
    {
        return $this->glueClient->deleteJob([
            'JobName' => $jobName,
        ]);
    }
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
```

```
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.

        :param job_name: The name of the job definition to delete.
        """
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteJobin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
```

```
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Einzelheiten zur API finden Sie DeleteJobin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Einzelheiten zur API finden Sie DeleteJobin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Delete an AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to delete.
    ///
    /// - Returns: `true` if the job is successfully deleted, otherwise `false`.
    func deleteJob(glueClient: GlueClient, name jobName: String) async -> Bool {
        do {
            _ = try await glueClient.deleteJob(
                input: DeleteJobInput(jobName: jobName)
            )
        } catch {
            return false
        }
        return true
    }
```

- Einzelheiten zur API finden Sie DeleteJobin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`DeleteTable`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `DeleteTable` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK für .NET API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const deleteTable = (databaseName, tableName) => {
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        echo "Delete the tables.\n";
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }

    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie DeleteTablein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
 table resides.
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Einzelheiten zur API finden Sie DeleteTablein der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
        }
```

- Einzelheiten zur API finden Sie DeleteTablein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
```

```
                        name: tableNames[0]
                    )
                )
            } catch {
                print("*** Unable to delete the table.")
            }
```

- Einzelheiten zur API finden Sie DeleteTablein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetCrawler`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `GetCrawler` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Get information about an AWS Glue crawler.
/// </summary>
/// <param name="crawlerName">The name of the crawler.</param>
/// <returns>A Crawler object describing the crawler.</returns>
public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
{
```

```
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
found.");
        return null;
    }
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetCrawlerRequest request;
        request.SetName(CRAWLER_NAME);
```

```
        Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

        if (outcome.IsSuccess()) {
            Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
            std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                              crawlerState)
                      << "." << std::endl;
        }
        else {
            std::cerr << "Error retrieving a crawler.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /**
     * Retrieves a specific crawler from the AWS Glue service and waits for it to
be in the "READY" state.
     *
     * @param glueClient  the AWS Glue client used to interact with the Glue
service
     * @param crawlerName the name of the crawler to be retrieved
     */
    public static void getSpecificCrawler(GlueClient glueClient, String
crawlerName) throws InterruptedException {
```

```
        try {
            GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            boolean ready = false;
            while (!ready) {
                GetCrawlerResponse response =
 glueClient.getCrawler(crawlerRequest);
                String status = response.crawler().stateAsString();
                if (status.compareTo("READY") == 0) {
                    ready = true;
                }
                Thread.sleep(3000);
            }

            System.out.println("The crawler is now ready");

        } catch (GlueException | InterruptedException e) {
            throw e;
        }
    }
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });
```

```
  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun getSpecificCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
        }
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}
```

- Einzelheiten zur API finden Sie GetCrawlerin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
    }
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
            crawler = response["Crawler"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return crawler
```

- Einzelheiten zur API finden Sie GetCrawlerin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
 if not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie GetCrawlerin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
            let tmp_crawler = glue
                .get_crawler()
                .name(self.crawler())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
```

- Einzelheiten zur API finden Sie GetCrawlerin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Get the state of the specified AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler whose state should be returned.
    ///
    /// - Returns: A `GlueClientTypes.CrawlerState` value describing the
```

```
    ///   state of the crawler.
    func getCrawlerState(glueClient: GlueClient, name: String) async ->
GlueClientTypes.CrawlerState {
        do {
            let output = try await glueClient.getCrawler(
                input: GetCrawlerInput(name: name)
            )

            // If the crawler or its state is `nil`, report that the crawler
            // is stopping. This may not be what you want for your
            // application but it works for this one!

            guard let crawler = output.crawler else {
                return GlueClientTypes.CrawlerState.stopping
            }
            guard let state = crawler.state else {
                return GlueClientTypes.CrawlerState.stopping
            }
            return state
        } catch {
            return GlueClientTypes.CrawlerState.stopping
        }
    }
```

- Einzelheiten zur API finden Sie GetCrawlerin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetDatabase`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `GetDatabase` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetDatabaseRequest request;
        request.SetName(CRAWLER_DATABASE_NAME);

        Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

        if (outcome.IsSuccess()) {
            const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

            std::cout << "Successfully retrieve the database\n" <<
                      database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
        }
        else {
            std::cerr << "Error getting the database.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK für C++ API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Retrieves the specific database from the AWS Glue service.
    *
    * @param glueClient   an instance of the AWS Glue client used to interact
with the service
    * @param databaseName the name of the database to retrieve
    * @throws GlueException if there is an error retrieving the database from
the AWS Glue service
    */
   public static void getSpecificDatabase(GlueClient glueClient, String
databaseName) {
       try {
           GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
               .name(databaseName)
               .build();

           GetDatabaseResponse response =
glueClient.getDatabase(databasesRequest);
           Instant createDate = response.database().createTime();

           // Convert the Instant to readable date.
           DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
               .withLocale(Locale.US)
               .withZone(ZoneId.systemDefault());

           formatter.format(createDate);
           System.out.println("The create date of the database is " +
createDate);

       } catch (GlueException e) {
           throw e;
       }
   }
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getDatabase = (name) => {
  const client = new GlueClient({});

  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun getSpecificDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
```

```
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}
```

- Einzelheiten zur API finden Sie GetDatabasein der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $databaseName = "doc-example-database-$uniqid";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

    public function getDatabase(string $databaseName): Result
    {
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["Database"]
```

- Einzelheiten zur API finden Sie GetDatabasein AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
 nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get database #{name}: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie GetDatabasein der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let database = glue
            .get_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?
            .to_owned();
        let database = database
            .database()
            .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;
```

- Einzelheiten zur API finden Sie GetDatabasein der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Get the AWS Glue database with the specified name.
    ///
```

```
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the database to return.
    ///
    /// - Returns: The `GlueClientTypes.Database` object describing the
    ///   specified database, or `nil` if an error occurs or the database
    ///   isn't found.
    func getDatabase(glueClient: GlueClient, name: String) async ->
GlueClientTypes.Database? {
        do {
            let output = try await glueClient.getDatabase(
                input: GetDatabaseInput(name: name)
            )

            return output.database
        } catch {
            return nil
        }
    }
```

- Einzelheiten zur API finden Sie GetDatabasein der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetDatabases`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetDatabases` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

CLI

AWS CLI

Um die Definitionen einiger oder aller Datenbanken im AWS Glue-Datenkatalog aufzulisten

Das folgende Beispiel für `get-databases` gibt Informationen über die Datenbanken im Datenkatalog zurück.

```
aws glue get-databases
```

Ausgabe:

```
{
    "DatabaseList": [
        {
            "Name": "default",
            "Description": "Default Hive database",
            "LocationUri": "file:/spark-warehouse",
            "CreateTime": 1602084052.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "flights-db",
            "CreateTime": 1587072847.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "legislators",
```

```
            "CreateTime": 1601415625.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "tempdb",
            "CreateTime": 1601498566.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        }
    ]
}
```

Weitere Informationen finden Sie unter Definieren einer Datenbank in Ihrem Datenkatalog im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie GetDatabasesin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getDatabases = () => {
  const client = new GlueClient({});

  const command = new GetDatabasesCommand({});

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetDatabasesin der AWS SDK für JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetJob`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetJob` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

CLI

AWS CLI

Informationen zu einem Auftrag abrufen

Das folgende Beispiel für `get-job` ruft Informationen zu einem Auftrag ab.

```
aws glue get-job \
    --job-name my-testing-job
```

Ausgabe:

```
{
    "Job": {
        "Name": "my-testing-job",
        "Role": "Glue_DefaultRole",
        "CreatedOn": 1602805698.167,
        "LastModifiedOn": 1602805698.167,
        "ExecutionProperty": {
            "MaxConcurrentRuns": 1
        },
        "Command": {
            "Name": "gluestreaming",
            "ScriptLocation": "s3://janetst-bucket-01/Scripts/test_script.scala",
            "PythonVersion": "2"
        },
        "DefaultArguments": {
            "--class": "GlueApp",
            "--job-language": "scala"
        },
        "MaxRetries": 0,
        "AllocatedCapacity": 10,
        "MaxCapacity": 10.0,
        "GlueVersion": "1.0"
    }
}
```

Weitere Informationen finden Sie unter Aufträge im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie GetJobin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getJob = (jobName) => {
  const client = new GlueClient({});

  const command = new GetJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetJobin der AWS SDK für JavaScript API-Referenz.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetJobRun`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetJobRun` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
{ JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }
```

- Einzelheiten zur API finden Sie GetJobRunin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
```

```
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
```

- Einzelheiten zur API finden Sie GetJobRunin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Informationen zu einer Auftragsausführung abrufen

Das folgende Beispiel für `get-job-run` ruft Informationen zu einer Auftragsausführung ab.

```
aws glue get-job-run \
    --job-name "Combine legistators data" \
    --run-id jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e
```

Ausgabe:

```
{
```

```
    "JobRun": {
        "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
        "Attempt": 0,
        "JobName": "Combine legistators data",
        "StartedOn": 1602873931.255,
        "LastModifiedOn": 1602874075.985,
        "CompletedOn": 1602874075.985,
        "JobRunState": "SUCCEEDED",
        "Arguments": {
            "--enable-continuous-cloudwatch-log": "true",
            "--enable-metrics": "",
            "--enable-spark-ui": "true",
            "--job-bookmark-option": "job-bookmark-enable",
            "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
        },
        "PredecessorRuns": [],
        "AllocatedCapacity": 10,
        "ExecutionTime": 117,
        "Timeout": 2880,
        "MaxCapacity": 10.0,
        "WorkerType": "G.1X",
        "NumberOfWorkers": 10,
        "LogGroupName": "/aws-glue/jobs",
        "GlueVersion": "2.0"
    }
}
```

Weitere Informationen finden Sie unter Auftragsausführungen im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie GetJobRunin der AWS CLI Befehlsreferenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetJobRunin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $jobName = 'test-job-' . $uniqid;

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
 $outputBucketUrl)['JobRunId'];

        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
 ['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

    public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
 Result
    {
```

```
        return $this->glueClient->getJobRun([
            'JobName' => $jobName,
            'RunId' => $runId,
            'PredecessorsIncluded' => $predecessorsIncluded,
        ]);
    }
```

- Einzelheiten zur API finden Sie GetJobRunin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
```

```
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRun"]
```

- Einzelheiten zur API finden Sie GetJobRunin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
```

```
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Einzelheiten zur API finden Sie GetJobRunin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let get_job_run = || async {
            Ok::<JobRun, GlueMvpError>(
                glue.get_job_run()
                    .job_name(self.job())
                    .run_id(job_run_id.to_string())
                    .send()
                    .await
                    .map_err(GlueMvpError::from_glue_sdk)?
                    .job_run()
                    .ok_or_else(|| GlueMvpError::Unknown("Failed to get
job_run".into()))?
                    .to_owned(),
            )
        };

        let mut job_run = get_job_run().await?;
        let mut state =
job_run.job_run_state().unwrap_or(&unknown_state).to_owned();

        while matches!(
            state,
            JobRunState::Starting | JobRunState::Stopping | JobRunState::Running
        ) {
            info!(?state, "Waiting for job to finish");
```

```
            tokio::time::sleep(self.wait_delay).await;

            job_run = get_job_run().await?;
            state = job_run.job_run_state().unwrap_or(&unknown_state).to_owned();
        }
```

- Einzelheiten zur API finden Sie GetJobRunin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Get information about a specific AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to return job run data for.
    ///   - id: The run ID of the specific job run to return.
    ///
    /// - Returns: A `GlueClientTypes.JobRun` object describing the state of
    ///   the job run, or `nil` if an error occurs.
    func getJobRun(glueClient: GlueClient, name jobName: String, id: String)
 async -> GlueClientTypes.JobRun? {
        do {
            let output = try await glueClient.getJobRun(
                input: GetJobRunInput(
                    jobName: jobName,
                    runId: id
                )
            )

            return output.jobRun
```

```
        } catch {
            return nil
        }
    }
```

- Einzelheiten zur API finden Sie GetJobRunin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetJobRuns`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetJobRuns` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Get information about all AWS Glue runs of a specific job.
/// </summary>
/// <param name="jobName">The name of the job.</param>
/// <returns>A list of JobRun objects.</returns>
public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
{
    var jobRuns = new List<JobRun>();
```

```
        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };

        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);
```

```
        Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
        getJobRunsRequest.SetJobName(jobName);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::Glue::Model::JobRun> allJobRuns;
        do {
            if (!nextToken.empty()) {
                getJobRunsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                    getJobRunsRequest);

            if (jobRunsOutcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
                allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

                nextToken = jobRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error getting job runs. "
                          << jobRunsOutcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!nextToken.empty());
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Informationen über alle Ausführungen eines Auftrags abrufen

Das folgende Beispiel für `get-job-runs` ruft Informationen zu allen Ausführungen eines Auftrags ab.

```
aws glue get-job-runs \
```

```
    --job-name "my-testing-job"
```

Ausgabe:

```
{
    "JobRuns": [
        {
            "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
            "Attempt": 0,
            "JobName": "my-testing-job",
            "StartedOn": 1602873931.255,
            "LastModifiedOn": 1602874075.985,
            "CompletedOn": 1602874075.985,
            "JobRunState": "SUCCEEDED",
            "Arguments": {
                "--enable-continuous-cloudwatch-log": "true",
                "--enable-metrics": "",
                "--enable-spark-ui": "true",
                "--job-bookmark-option": "job-bookmark-enable",
                "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
            },
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 117,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_2",
            "Attempt": 2,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "JobName": "my-testing-job",
            "StartedOn": 1602811168.496,
            "LastModifiedOn": 1602811282.39,
            "CompletedOn": 1602811282.39,
```

```
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 021AAB703DB20A2D;
                S3 Extended Request ID: teZk24Y09TkXzBvMPG502L5VJBhe9DJuWA9/
TXtuGOqfByajkfL/Tlqt5JBGdEGpigAqzdMDM/U=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 110,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "Attempt": 1,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f",
            "JobName": "my-testing-job",
            "StartedOn": 1602811020.518,
            "LastModifiedOn": 1602811138.364,
            "CompletedOn": 1602811138.364,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 2671D37856AE7ABB;
                S3 Extended Request ID: RLJCJw20brV
+PpC6GpORahyF2fp9flB5SSb2bTGPnUSPVizLXRl1PN3QZldb+v1o9qRVktNYbW8=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 113,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
```

```
            "GlueVersion": "2.0"
        }
    ]
}
```

Weitere Informationen finden Sie unter Auftragsausführungen im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie GetJobRunsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Retrieves the job runs for a given Glue job and prints the status of the
job runs.
    *
    * @param glueClient the Glue client used to make API calls
    * @param jobName    the name of the Glue job to retrieve the job runs for
    */
   public static void getJobRuns(GlueClient glueClient, String jobName) {
       try {
           GetJobRunsRequest runsRequest = GetJobRunsRequest.builder()
               .jobName(jobName)
               .maxResults(20)
               .build();

           boolean jobDone = false;
           while (!jobDone) {
               GetJobRunsResponse response = glueClient.getJobRuns(runsRequest);
               List<JobRun> jobRuns = response.jobRuns();
               for (JobRun jobRun : jobRuns) {
                   String jobState = jobRun.jobRunState().name();
                   if (jobState.compareTo("SUCCEEDED") == 0) {
                       System.out.println(jobName + " has succeeded");
```

```
                        jobDone = true;

                    } else if (jobState.compareTo("STOPPED") == 0) {
                        System.out.println("Job run has stopped");
                        jobDone = true;

                    } else if (jobState.compareTo("FAILED") == 0) {
                        System.out.println("Job run has failed");
                        jobDone = true;

                    } else if (jobState.compareTo("TIMEOUT") == 0) {
                        System.out.println("Job run has timed out");
                        jobDone = true;

                    } else {
                        System.out.println("*** Job run state is " +
jobRun.jobRunState().name());
                        System.out.println("Job run Id is " + jobRun.id());
                        System.out.println("The Glue version is " +
jobRun.glueVersion());
                    }
                    TimeUnit.SECONDS.sleep(5);
                }
            }

        } catch (GlueException e) {
            throw e;
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getJobRuns = (jobName) => {
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $jobName = 'test-job-' . $uniqid;

        $jobRuns = $glueService->getJobRuns($jobName);

    public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
 Result
    {
        $arguments = ['JobName' => $jobName];
```

```
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        return $this->glueClient->getJobRuns($arguments);
    }
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
        :return: The list of job runs.
        """
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
```

```
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]
```

- Einzelheiten zur API finden Sie GetJobRunsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
```

```
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Einzelheiten zur API finden Sie GetJobRunsin der AWS SDK für Ruby API-Referenz.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Return a list of the job runs for the specified job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job for which to return its job runs.
    ///   - maxResults: The maximum number of job runs to return (default:
    ///     1000).
    ///
    /// - Returns: An array of `GlueClientTypes.JobRun` objects describing
    ///   each job run.
    func getJobRuns(glueClient: GlueClient, name jobName: String, maxResults:
 Int? = nil) async -> [GlueClientTypes.JobRun] {
        do {
            let output = try await glueClient.getJobRuns(
                input: GetJobRunsInput(
                    jobName: jobName,
                    maxResults: maxResults
                )
            )
```

```
            guard let jobRuns = output.jobRuns else {
                print("*** No job runs found.")
                return []
            }

            return jobRuns
        } catch is EntityNotFoundException {
            print("*** The specified job name, \(jobName), doesn't exist.")
            return []
        } catch {
            print("*** Unexpected error getting job runs:")
            dump(error)
            return []
        }
    }
```

- Einzelheiten zur API finden Sie GetJobRunsin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`GetTables`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetTables` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Get a list of tables for an AWS Glue database.
/// </summary>
/// <param name="dbName">The name of the database.</param>
/// <returns>A list of Table objects.</returns>
public async Task<List<Table>> GetTablesAsync(string dbName)
{
    var request = new GetTablesRequest { DatabaseName = dbName };
    var tables = new List<Table>();

    // Get a paginator for listing the tables.
    var tablePaginator = _amazonGlue.Paginators.GetTables(request);

    await foreach (var response in tablePaginator.Responses)
    {
        tables.AddRange(response.TableList);
    }

    return tables;
}
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::GetTablesRequest request;
       request.SetDatabaseName(CRAWLER_DATABASE_NAME);
       std::vector<Aws::Glue::Model::Table> all_tables;
       Aws::String nextToken; // Used for pagination.
       do {
           Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

           if (outcome.IsSuccess()) {
               const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
               all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
               nextToken = outcome.GetResult().GetNextToken();
           }
           else {
               std::cerr << "Error getting the tables. "
                         << outcome.GetError().GetMessage()
                         << std::endl;
               deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                            clientConfig);
               return false;
           }
       } while (!nextToken.empty());
```

```
        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Die Definitionen einiger oder aller Tabellen in der angegebenen Datenbank auflisten

Das folgende Beispiel für `get-tables` gibt Informationen zu den Tabellen in der angegebenen Datenbank zurück.

```
aws glue get-tables --database-name 'tempdb'
```

Ausgabe:

```
{
    "TableList": [
        {
            "Name": "my-s3-sink",
            "DatabaseName": "tempdb",
```

```
            "CreateTime": 1602730539.0,
            "UpdateTime": 1602730539.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/test-s3-output/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "s3-source",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730658.0,
            "UpdateTime": 1602730658.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
```

```
                    "Type": "int"
                },
                {
                    "Name": "currenttemperature",
                    "Type": "int"
                },
                {
                    "Name": "status",
                    "Type": "string"
                }
            ],
            "Location": "s3://janetst-bucket-01/",
            "Compressed": false,
            "NumberOfBuckets": 0,
            "SortColumns": [],
            "StoredAsSubDirectories": false
        },
        "Parameters": {
            "classification": "json"
        },
        "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
        "IsRegisteredWithLakeFormation": false,
        "CatalogId": "007436865787"
    },
    {
        "Name": "test-kinesis-input",
        "DatabaseName": "tempdb",
        "CreateTime": 1601507001.0,
        "UpdateTime": 1601507001.0,
        "Retention": 0,
        "StorageDescriptor": {
            "Columns": [
                {
                    "Name": "sensorid",
                    "Type": "int"
                },
                {
                    "Name": "currenttemperature",
                    "Type": "int"
                },
                {
                    "Name": "status",
                    "Type": "string"
                }
```

```
                ],
                "Location": "my-testing-stream",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "Parameters": {
                    "kinesisUrl": "https://kinesis.us-east-1.amazonaws.com",
                    "streamName": "my-testing-stream",
                    "typeOfData": "kinesis"
                },
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        }
    ]
}
```

Weitere Informationen finden Sie unter Definieren von Tabellen im AWS Glue-Datenkatalog im AWS Glue-Entwicklerhandbuch.

- Einzelheiten zur API finden Sie GetTablesin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/**
```

```
    * Retrieves the names of the tables in the specified Glue database.
    *
    * @param glueClient the Glue client to use for the operation
    * @param dbName     the name of the Glue database to retrieve the table
names from
    * @return the name of the first table retrieved, or an empty string if no
tables were found
    */
   public static String getGlueTables(GlueClient glueClient, String dbName) {
       String myTableName = "";
       try {
           GetTablesRequest tableRequest = GetTablesRequest.builder()
               .databaseName(dbName)
               .build();

           GetTablesResponse response = glueClient.getTables(tableRequest);
           List<Table> tables = response.tableList();
           if (tables.isEmpty()) {
               System.out.println("No tables were returned");
           } else {
               for (Table table : tables) {
                   myTableName = table.name();
                   System.out.println("Table name is: " + myTableName);
               }
           }

       } catch (GlueException e) {
           throw e;
       }
       return myTableName;
   }
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);

    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
```

```
            'DatabaseName' => $databaseName,
        ]);
    }
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
```

```
            )
            raise
        else:
            return response["TableList"]
```

- Einzelheiten zur API finden Sie GetTablesin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
```

```
  end
```

- Einzelheiten zur API finden Sie GetTablesin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let tables = glue
            .get_tables()
            .database_name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

- Einzelheiten zur API finden Sie GetTablesin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
```

```
import AWSGlue

    /// Returns a list of the tables in the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database whose tables are to be
    ///     returned.
    ///
    /// - Returns: An array of `GlueClientTypes.Table` objects, each
    ///   describing one table in the named database. An empty array indicates
    ///   that there are either no tables in the database, or an error
    ///   occurred before any tables could be found.
    func getTablesInDatabase(glueClient: GlueClient, databaseName: String) async
 -> [GlueClientTypes.Table] {
        var tables: [GlueClientTypes.Table] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.getTables(
                    input: GetTablesInput(
                        databaseName: databaseName,
                        nextToken: nextToken
                    )
                )

                guard let tableList = output.tableList else {
                    return tables
                }

                tables = tables + tableList
                nextToken = output.nextToken
            } catch {
                return tables
            }
        } while nextToken != nil

        return tables
    }
```

- Einzelheiten zur API finden Sie GetTablesin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`ListJobs`** mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `ListJobs` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::ListJobsRequest listJobsRequest;

        Aws::String nextToken;
        std::vector<Aws::String> allJobNames;

        do {
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
```

```
                        << listRunsOutcome.GetError().GetMessage()
                        << std::endl;
            }
        } while (!nextToken.empty());
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für C++ API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const listJobs = () => {
  const client = new GlueClient({});

  const command = new ListJobsCommand({});

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
            echo "{$jobsName}\n";
        }

    public function listJobs($maxResults = null, $nextToken = null, $tags = []):
 Result
    {
        $arguments = [];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        if (!empty($tags)) {
            $arguments['Tags'] = $tags;
        }
        return $this->glueClient->listJobs($arguments);
    }
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
```

```
        """
        self.glue_client = glue_client


    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobNames"]
```

- Einzelheiten zur API finden Sie ListJobsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
```

```
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie ListJobsin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Einzelheiten zur API finden Sie ListJobsin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Return a list of the AWS Glue jobs listed on the user's account.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - maxJobs: The maximum number of jobs to return (default: 100).
    ///
    /// - Returns: An array of strings listing the names of all available AWS
    ///   Glue jobs.
    func listJobs(glueClient: GlueClient, maxJobs: Int = 100) async -> [String] {
        var jobList: [String] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.listJobs(
                    input: ListJobsInput(
                        maxResults: maxJobs,
                        nextToken: nextToken
                    )
                )

                guard let jobs = output.jobNames else {
                    return jobList
                }
```

```
                jobList = jobList + jobs
                nextToken = output.nextToken
            } catch {
                return jobList
            }
        } while (nextToken != nil)

        return jobList
    }
```

- Einzelheiten zur API finden Sie ListJobsin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`StartCrawler`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `StartCrawler` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Start an AWS Glue crawler.
/// </summary>
/// <param name="crawlerName">The name of the crawler.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> StartCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new StartCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::StartCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


        if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
```

```
                                    outcome.GetError().GetErrorType())) {
            if (!outcome.IsSuccess()) {
                std::cout << "Crawler was already started." << std::endl;
            }
            else {
                std::cout << "Successfully started crawler." << std::endl;
            }

            std::cout << "This may take a while to run." << std::endl;

            Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
            int iterations = 0;
            while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
                std::this_thread::sleep_for(std::chrono::seconds(1));
                ++iterations;
                if ((iterations % 10) == 0) { // Log status every 10 seconds.
                    std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                      crawlerState)
                              << ". After " << iterations
                              << " seconds elapsed."
                              << std::endl;
                }
                Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
                getCrawlerRequest.SetName(CRAWLER_NAME);

                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }
```

```
            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
                          << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Einen Crawler starten

Das folgende Beispiel für `start-crawler` startet einen Crawler.

```
aws glue start-crawler --name my-crawler
```

Ausgabe:

```
None
```

Weitere Informationen finden Sie unter Definieren von Crawlern im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie StartCrawlerin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Starts a specific AWS Glue crawler.
    *
    * @param glueClient  the AWS Glue client to use for the crawler operation
    * @param crawlerName the name of the crawler to start
    * @throws GlueException if there is an error starting the crawler
    */
   public static void startSpecificCrawler(GlueClient glueClient, String
crawlerName) {
       try {
           StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           glueClient.startCrawler(crawlerRequest);
           System.out.println(crawlerName + " was successfully started!");

       } catch (GlueException e) {
           throw e;
       }
   }
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK für JavaScript API-Referenz.

Kotlin

SDK für Kotlin

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
suspend fun startSpecificCrawler(crawlerName: String?) {
    val request =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-west-2" }.use { glueClient ->
```

```
        glueClient.startCrawler(request)
        println("$crawlerName was successfully started.")
    }
}
```

- Einzelheiten zur API finden Sie StartCrawlerin der API-Referenz zum AWS SDK für Kotlin.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $databaseName = "doc-example-database-$uniqid";

        $glueService->startCrawler($crawlerName);

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Einzelheiten zur API finden Sie StartCrawlerin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie StartCrawlerin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

        match start_crawler {
            Ok(_) => Ok(()),
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
        }?;
```

- Einzelheiten zur API finden Sie StartCrawlerin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue
```

```
    /// Start running an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use when starting the crawler.
    ///   - name: The name of the crawler to start running.
    ///
    /// - Returns: `true` if the crawler is started successfully, otherwise
`false`.
    func startCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.startCrawler(
                input: StartCrawlerInput(name: name)
            )
        } catch {
            print("*** An unexpected error occurred starting the crawler.")
            return false
        }

        return true
    }
```

- Einzelheiten zur API finden Sie StartCrawlerin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwendung **`StartJobRun`** mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `StartJobRun` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erlernen der Grundlagen

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
/// <summary>
/// Start an AWS Glue job run.
/// </summary>
/// <param name="jobName">The name of the job.</param>
/// <returns>A string representing the job run Id.</returns>
public async Task<string> StartJobRunAsync(
    string jobName,
    string inputDatabase,
    string inputTable,
    string bucketName)
{
    var request = new StartJobRunRequest
    {
        JobName = jobName,
        Arguments = new Dictionary<string, string>
        {
            {"--input_database", inputDatabase},
            {"--input_table", inputTable},
            {"--output_bucket_url", $"s3://{bucketName}/"}
        }
    };

    var response = await _amazonGlue.StartJobRunAsync(request);
    return response.JobRunId;
}
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK für .NET API-Referenz.

C++

SDK für C++

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::StartJobRunRequest request;
       request.SetJobName(JOB_NAME);

       Aws::Map<Aws::String, Aws::String> arguments;
       arguments["--input_database"] = CRAWLER_DATABASE_NAME;
       arguments["--input_table"] = tableName;
       arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
       request.SetArguments(arguments);

       Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);

       if (outcome.IsSuccess()) {
           std::cout << "Successfully started the job." << std::endl;

           Aws::String jobRunId = outcome.GetResult().GetJobRunId();

           int iterator = 0;
           bool done = false;
           while (!done) {
               ++iterator;
               std::this_thread::sleep_for(std::chrono::seconds(1));
               Aws::Glue::Model::GetJobRunRequest jobRunRequest;
               jobRunRequest.SetJobName(JOB_NAME);
```

```
                jobRunRequest.SetRunId(jobRunId);

                Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                        jobRunRequest);

                if (jobRunOutcome.IsSuccess()) {
                    const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                    Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();

                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                         jobRunState) <<
                                  ". " << iterator <<
                                  " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
```

```
                            << jobRunOutcome.GetError().GetMessage()
                            << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK für C++ API-Referenz.

CLI

AWS CLI

Die Auftragsausführung starten

Das folgende Beispiel für `start-job-run` startet die Ausführung eines Auftrags.

```
aws glue start-job-run \
    --job-name my-job
```

Ausgabe:

```
{
    "JobRunId":
 "jr_22208b1f44eb5376a60569d4b21dd20fcb8621e1a366b4e7b2494af764b82ded"
}
```

Weitere Informationen finden Sie unter Autorisieren von Aufträgen im Entwicklerhandbuch für AWS Glue.

- Einzelheiten zur API finden Sie StartJobRunin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
   /**
    * Starts a job run in AWS Glue.
    *
    * @param glueClient    the AWS Glue client to use for the job run
    * @param jobName       the name of the Glue job to run
    * @param inputDatabase the name of the input database
    * @param inputTable    the name of the input table
    * @param outBucket     the URL of the output S3 bucket
    * @throws GlueException if there is an error starting the job run
    */
   public static void startJob(GlueClient glueClient, String jobName, String
inputDatabase, String inputTable,
                               String outBucket) {
       try {
           Map<String, String> myMap = new HashMap<>();
           myMap.put("--input_database", inputDatabase);
           myMap.put("--input_table", inputTable);
           myMap.put("--output_bucket_url", outBucket);

           StartJobRunRequest runRequest = StartJobRunRequest.builder()
               .workerType(WorkerType.G_1_X)
               .numberOfWorkers(10)
               .arguments(myMap)
               .jobName(jobName)
               .build();

           StartJobRunResponse response = glueClient.startJobRun(runRequest);
           System.out.println("The request Id of the job is " +
response.responseMetadata().requestId());

       } catch (GlueException e) {
```

```
            throw e;
        }
    }
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK for Java 2.x API-Referenz.

JavaScript

SDK für JavaScript (v3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK für JavaScript API-Referenz.

PHP

SDK für PHP

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        $jobName = 'test-job-' . $uniqid;

        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

    public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
    {
        return $this->glueClient->startJobRun([
            'JobName' => $jobName,
            'Arguments' => [
                'input_database' => $databaseName,
                'input_table' => $tables['TableList'][0]['Name'],
                'output_bucket_url' => $outputBucketUrl,
                '--input_database' => $databaseName,
                '--input_table' => $tables['TableList'][0]['Name'],
                '--output_bucket_url' => $outputBucketUrl,
            ],
        ]);
    }
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK für PHP API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_job_run(self, name, input_database, input_table,
 output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
 tables
                               that describe the source data. This is typically
 created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
 the
            # Python ETL script to determine the location of input and output
 data.
```

```
            response = self.glue_client.start_job_run(
                JobName=name,
                Arguments={
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRunId"]
```

- Einzelheiten zur API finden Sie StartJobRunin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
```

```
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end
```

- Einzelheiten zur API finden Sie StartJobRunin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
        let job_run_output = glue
            .start_job_run()
```

```
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

- Einzelheiten zur API finden Sie StartJobRunin der API-Referenz zum AWS SDK für Rust.

Swift

SDK für Swift

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import AWSClientRuntime
import AWSGlue

    /// Start an AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to run.
```

```
   ///   - databaseName: The name of the AWS Glue database to run the job
against.
   ///   - tableName: The name of the table in the database to run the job
against.
   ///   - outputURL: The AWS S3 URI of the bucket location into which to
   ///     write the resulting output.
   ///
   /// - Returns: `true` if the job run is started successfully, otherwise
`false`.
   func startJobRun(glueClient: GlueClient, name jobName: String, databaseName:
String,
                    tableName: String, outputURL: String) async -> String? {
       do {
           let output = try await glueClient.startJobRun(
               input: StartJobRunInput(
                   arguments: [
                       "--input_database": databaseName,
                       "--input_table": tableName,
                       "--output_bucket_url": outputURL
                   ],
                   jobName: jobName,
                   numberOfWorkers: 10,
                   workerType: .g1x
               )
           )

           guard let id = output.jobRunId else {
               return nil
           }

           return id
       } catch {
           return nil
       }
   }
```

- Einzelheiten zur API finden Sie StartJobRunin der API-Referenz zum AWS SDK für Swift.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterVerwenden dieses Dienstes mit einem AWS SDK. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Sicherheit in AWS Glue

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS -Compliance-Programme regelmäßig. Weitere Informationen zu den Compliance-Programmen, die für gelten AWS Glue, finden Sie unter AWS Services in Umfang nach Compliance-Programmen.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der geteilten Verantwortung bei der Verwendung von AWS Glue einsetzen können. Die folgenden Themen veranschaulichen, wie Sie AWS Glue zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie lernen auch, wie Sie andere AWS Dienste nutzen können, die Ihnen helfen, Ihre AWS Glue Ressourcen zu überwachen und zu sichern.

Themen

- Datenschutz in AWS Glue
- Identitäts- und Zugriffsmanagement für AWS Glue
- AWS Lake Formation Modelle für die Zugriffskontrolle
- Verwenden von Amazon S3 Access Grants mit AWS Glue
- Einloggen und Überwachen AWS Glue
- Konformitätsvalidierung für AWS Glue
- Resilienz in AWS Glue
- Infrastruktursicherheit in AWS Glue

Datenschutz in AWS Glue

AWS Glue bietet verschiedene Funktionen, die zum Schutz Ihrer Daten beitragen sollen.

Themen

- Verschlüsseln von Daten im Ruhezustand
- Verschlüsseln von Daten während der Übertragung
- Compliance mit FIPS
- Schlüsselverwaltung
- AWS Glue Abhängigkeit von anderen AWS Diensten
- Entwicklungsendpunkte

Verschlüsseln von Daten im Ruhezustand

AWS Glue unterstützt Datenverschlüsselung im Ruhezustand für Erstellung visueller ETL-Jobs undEntwickeln von Skripts unter Verwendung von Entwicklungsendpunkten. Sie können ETL-Aufträge (Extrahieren, Transformieren und Laden) sowie Entwicklungsendpunkte konfigurieren, um AWS Key Management Service (AWS KMS)-Schlüssel zum Schreiben verschlüsselter Daten im Ruhezustand zu verwenden. Sie können auch die Metadaten verschlüsseln, die in den AWS Glue Data Catalog von Ihnen verwalteten Schlüsseln gespeichert sind. AWS KMS Darüber hinaus können Sie AWS KMS Schlüssel verwenden, um Job-Lesezeichen und die von Crawlern und ETL-Jobs generierten Logs zu verschlüsseln.

Sie können Metadatenobjekte AWS Glue Data Catalog in Ihrem zusätzlich zu den Daten verschlüsseln, die von Jobs, Crawlern und Entwicklungsendpunkten in Amazon Simple Storage Service (Amazon S3) und Amazon CloudWatch Logs geschrieben wurden. Wenn Sie Jobs, Crawler und Entwicklungsendpunkte in erstellen AWS Glue, können Sie Verschlüsselungseinstellungen angeben, indem Sie eine Sicherheitskonfiguration anhängen. Sicherheitskonfigurationen enthalten von Amazon S3 verwaltete serverseitige Verschlüsselungsschlüssel (SSE-S3) oder Kunden-Masterschlüssel (), die in (SSE-KMSCMKs) gespeichert sind. AWS KMS Sie können Sicherheitskonfigurationen mit dem AWS Glue console.

Sie können die Verschlüsselung des gesamten Data Catalogs auch in Ihrem Konto aktivieren. Sie tun dies, indem Sie CMKs gespeichert in angeben AWS KMS.

> **Important**
>
> AWS Glue unterstützt nur symmetrische, vom Kunden verwaltete Schlüssel. Weitere Informationen finden Sie unter Vom Kunden verwaltete Schlüssel (CMKs) im AWS Key Management Service Entwicklerhandbuch.

Wenn die Verschlüsselung eingeschaltet ist, werden SSE-S3- oder SSE-KMS-Schlüssel verwendet, um Daten im Ruhezustand zu schreiben, wenn Sie Data-Catalog-Objekte hinzufügen, Crawler ausführen, Aufträge ausführen oder Entwicklungsendpunkte starten. Darüber hinaus können Sie Folgendes konfigurieren AWS Glue um nur über ein vertrauenswürdiges Transport Layer Security (TLS) -Protokoll auf Datenspeicher von Java Database Connectivity (JDBC) zuzugreifen.

In AWS Glue, Sie steuern die Verschlüsselungseinstellungen an den folgenden Stellen:

- Die Einstellungen von Data Catalog
- Die Sicherheitskonfigurationen, die Sie erstellen.
- Die serverseitige Verschlüsselungseinstellung (SSE-S3 oder SSE-KMS), die als Parameter an Ihren übergeben wird AWS Glue ETL-Job (Extrahieren, Transformieren und Laden).

Weitere Informationen zum Einrichten Ihrer Verschlüsselung finden Sie unter Einrichten der Verschlüsselung in AWS Glue.

Themen

- Verschlüsseln Ihres Data Catalog
- Verschlüsselung von Verbindungspasswörtern
- Verschlüsseln von Daten, die von AWS Glue geschrieben werden

Verschlüsseln Ihres Data Catalog

AWS Glue Data Catalog Die Verschlüsselung bietet mehr Sicherheit für Ihre sensiblen Daten. AWS Glue integriert sich in AWS Key Management Service (AWS KMS), um Metadaten zu verschlüsseln, die im Datenkatalog gespeichert sind. Sie können die Verschlüsselungseinstellungen für Ressourcen im Datenkatalog mithilfe der AWS Glue Konsole oder der AWS CLI aktivieren oder deaktivieren.

Wenn Sie die Verschlüsselung für Ihren Datenkatalog aktivieren, werden alle neuen Objekte, die Sie erstellen, verschlüsselt. Wenn Sie die Verschlüsselung deaktivieren, werden die neuen Objekte, die Sie erstellen, nicht verschlüsselt, aber vorhandene verschlüsselte Objekte bleiben verschlüsselt.

Sie können Ihren gesamten Datenkatalog mit AWS verwalteten oder vom Kunden verwalteten Verschlüsselungsschlüsseln verschlüsseln. Weitere Informationen zu Schlüsseltypen und Status finden Sie unter AWS Key Management Service Konzepte im AWS Key Management Service Entwicklerhandbuch.

> **Note**
>
> Wenn Sie den verschlüsselten Datenkatalog mit einem Crawler verwenden, müssen Sie die Verschlüsselungseinstellungen beibehalten. Das Entfernen von Verschlüsselungseinstellungen, nachdem ein Crawler einen verschlüsselten Katalog verarbeitet hat, führt zu Fehlern. Wenn Sie Verschlüsselungseinstellungen entfernen müssen, erstellen Sie einen neuen Crawler, anstatt den vorhandenen zu ändern.

AWS verwaltete Schlüssel

AWS Verwaltete Schlüssel sind KMS-Schlüssel in Ihrem Konto, die in Ihrem Namen von einem integrierten AWS Dienst erstellt, verwaltet und verwendet werden. AWS KMS Sie können die AWS verwalteten Schlüssel in Ihrem Konto einsehen, ihre wichtigsten Richtlinien einsehen und ihre Verwendung in AWS CloudTrail Protokollen überprüfen. Sie können diese Schlüssel jedoch nicht verwalten oder ihre Berechtigungen ändern.

Encryption at Rest wird automatisch in AWS KMS die Verwaltung der AWS verwalteten Schlüssel integriert AWS Glue , die zur Verschlüsselung Ihrer Metadaten verwendet werden. Wenn bei der Aktivierung der Metadatenverschlüsselung kein AWS verwalteter Schlüssel vorhanden ist, AWS KMS wird automatisch ein neuer Schlüssel für Sie erstellt.

Weitere Informationen finden Sie unter Von AWS verwaltete Schlüssel.

Kundenverwaltete Schlüssel

Kundenverwaltete Schlüssel sind KMS-Schlüssel in Ihrem AWS-Konto System, die Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über diese KMS-Schlüssel. Sie haben folgende Möglichkeiten:

- Richten Sie ihre wichtigsten Richtlinien, IAM-Richtlinien und Zuschüsse ein und pflegen Sie sie

- Aktiviere und deaktiviere sie
- Dreht ihr kryptografisches Material
- Tags hinzufügen
- Erstellen Sie Aliase, die auf sie verweisen
- Planen Sie, dass sie gelöscht werden

Weitere Informationen zur Verwaltung der Berechtigungen eines vom Kunden verwalteten Schlüssels finden Sie unter Vom Kunden verwaltete Schlüssel.

> **Important**
>
> AWS Glue unterstützt nur symmetrische, vom Kunden verwaltete Schlüssel. In der KMS-Schlüsselliste werden nur symmetrische Schlüssel angezeigt. Wenn Sie jedoch „KMS-Schlüssel-ARN auswählen“ auswählen, können Sie in der Konsole einen ARN für einen beliebigen Schlüsseltyp eingeben. Stellen Sie sicher, dass Sie nur ARNs für symmetrische Schlüssel eingeben.
> Um einen symmetrischen, vom Kunden verwalteten Schlüssel zu erstellen, folgen Sie den Schritten zur Erstellung symmetrischer, vom Kunden verwalteter Schlüssel im AWS Key Management Service Entwicklerhandbuch.

Wenn Sie die Datenkatalogverschlüsselung im Ruhezustand aktivieren, werden die folgenden Ressourcentypen mit KMS-Schlüsseln verschlüsselt:

- Datenbanken
- Tabellen
- Partitionen
- Tabellenversionen
- Spaltenstatistiken
- Benutzerdefinierte Funktionen
- Datenkatalog-Ansichten

AWS Glue Verschlüsselungskontext

Ein Verschlüsselungskontext ist ein optionaler Satz von Schlüssel-Wert-Paaren, der zusätzliche Kontextinformationen zu den Daten enthalten kann. AWS KMS verwendet den Verschlüsselungskontext als Additional Authenticated Data (AAD) zur Unterstützung der authentifizierten Verschlüsselung. Wenn Sie einen Verschlüsselungskontext in eine Anforderung zum Verschlüsseln von Daten einbeziehen, wird der Verschlüsselungskontext an die verschlüsselten Daten AWS KMS gebunden. Um Daten zu entschlüsseln, fügen Sie denselben Verschlüsselungskontext in die Anforderung ein. AWS Glue verwendet bei allen AWS KMS kryptografischen Vorgängen denselben Verschlüsselungskontext, wobei der Schlüssel `glue_catalog_id` und der Wert der ist. `catalogId`

```
"encryptionContext": {
    "glue_catalog_id": "111122223333"
}
```

Wenn Sie einen AWS verwalteten Schlüssel oder einen symmetrischen, vom Kunden verwalteten Schlüssel zur Verschlüsselung Ihres Datenkatalogs verwenden, können Sie den Verschlüsselungskontext auch in Prüfaufzeichnungen und Protokollen verwenden, um festzustellen, wie der Schlüssel verwendet wird. Der Verschlüsselungskontext erscheint auch in Protokollen, die von AWS CloudTrail oder Amazon CloudWatch Logs generiert werden.

Verschlüsselung aktivieren

Sie können die Verschlüsselung für Ihre AWS Glue Data Catalog Objekte in den Datenkatalogeinstellungen der AWS Glue Konsole aktivieren oder indem Sie den verwenden AWS CLI.

Console

So aktivieren Sie die Verschlüsselung über die Konsole

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Datenkatalog aus.
3. Aktivieren Sie auf der Seite mit den Datenkatalogeinstellungen das Kontrollkästchen Metadatenverschlüsselung und wählen Sie einen AWS KMS Schlüssel aus.

Wenn Sie die Verschlüsselung aktivieren und keinen vom Kunden verwalteten Schlüssel angeben, verwenden die Verschlüsselungseinstellungen einen AWS verwalteten KMS-Schlüssel.

4. (Optional) Wenn Sie einen vom Kunden verwalteten Schlüssel zum Verschlüsseln Ihres Datenkatalogs verwenden, bietet der Datenkatalog die Option, eine IAM-Rolle zum Verschlüsseln und Entschlüsseln von Ressourcen zu registrieren. Sie müssen Ihrer IAM-Rolle Berechtigungen erteilen, die Sie in Ihrem Namen übernehmen AWS Glue können. Dazu gehören auch AWS KMS Berechtigungen zum Verschlüsseln und Entschlüsseln von Daten.

   Wenn Sie eine neue Ressource im Datenkatalog erstellen, AWS Glue übernimmt sie die IAM-Rolle, die für die Verschlüsselung der Daten vorgesehen ist. Ähnlich verhält es sich, wenn ein Verbraucher auf die Ressource zugreift, AWS Glue übernimmt er die IAM-Rolle zur Entschlüsselung von Daten. Wenn Sie eine IAM-Rolle mit den erforderlichen Berechtigungen registrieren, benötigt der aufrufende Principal keine Berechtigungen mehr, um auf den Schlüssel zuzugreifen und die Daten zu entschlüsseln.

   > **Important**
   >
   > Sie können KMS-Operationen nur dann an eine IAM-Rolle delegieren, wenn Sie einen vom Kunden verwalteten Schlüssel zum Verschlüsseln der Datenkatalogressourcen verwenden. Die KMS-Rollendelegierungsfunktion unterstützt derzeit nicht die Verwendung AWS verwalteter Schlüssel zum Verschlüsseln von Datenkatalogressourcen.

   > **Warning**
   >
   > Wenn Sie eine IAM-Rolle zum Delegieren von KMS-Vorgängen aktivieren, können Sie nicht mehr auf die Datenkatalogressourcen zugreifen, die zuvor mit einem AWS verwalteten Schlüssel verschlüsselt wurden.

   a. Um eine IAM-Rolle zu aktivieren, die davon ausgehen AWS Glue kann, Daten in Ihrem Namen zu ver- und entschlüsseln, wählen Sie die Option KMS-Operationen an eine IAM-Rolle delegieren.

   b. Wählen Sie als Nächstes eine IAM-Rolle aus.

Eine Anleitung zum Erstellen einer IAM;-Rolle finden Sie unter Erstellen von IAM-Rollen für AWS Glue.

Die IAM-Rolle, die den Zugriff auf den Datenkatalog AWS Glue voraussetzt, muss über die Berechtigungen zum Verschlüsseln und Entschlüsseln von Metadaten im Datenkatalog verfügen. Sie können eine IAM-Rolle erstellen und die folgenden Inline-Richtlinien anhängen:

- Fügen Sie die folgende Richtlinie hinzu, um AWS KMS Berechtigungen zum Verschlüsseln und Entschlüsseln des Datenkatalogs einzubeziehen.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:Encrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:<region>:<account-id>:key/<key-id>"
        }
    ]
}
```

- Fügen Sie als Nächstes die folgende Vertrauensrichtlinie zur Rolle hinzu, damit der AWS Glue Dienst die IAM-Rolle übernimmt.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
```

```
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

- Fügen Sie als Nächstes die `iam:PassRole` Berechtigung zur IAM-Rolle hinzu.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<account-id>:role/<encryption-role-name>"
            ]
        }
    ]
}
```

Wenn Sie die Verschlüsselung aktivieren und keine IAM-Rolle für die Übernahme angegeben haben, muss der Principal, der AWS Glue auf den Datenkatalog zugreift, über die erforderlichen Berechtigungen verfügen, um die folgenden API-Operationen auszuführen:

- `kms:Decrypt`
- `kms:Encrypt`
- `kms:GenerateDataKey`

AWS CLI

Um die Verschlüsselung mit dem SDK zu aktivieren oder AWS CLI

- Verwenden Sie die API-Operation `PutDataCatalogEncryptionSettings`. Wenn kein Schlüssel angegeben ist, wird der AWS verwaltete Verschlüsselungsschlüssel für das Kundenkonto AWS Glue verwendet, um den Datenkatalog zu verschlüsseln.

```
aws glue put-data-catalog-encryption-settings \
  --data-catalog-encryption-settings '{
      "EncryptionAtRest": {
      "CatalogEncryptionMode": "SSE-KMS-WITH-SERVICE-ROLE",
      "SseAwsKmsKeyId": "arn:aws:kms:<region>:<account-id>:key/<key-id>",
      "CatalogEncryptionServiceRole":"arn:aws:iam::<account-
id>:role/<encryption-role-name>"
    }

  }'
```

Wenn Sie die Verschlüsselung aktivieren, werden alle Objekte, die Sie in den Datenkatalogobjekten erstellen, verschlüsselt. Wenn Sie diese Einstellung deaktivieren, werden die Objekte, die Sie im Datenkatalog erstellen, nicht mehr verschlüsselt. Sie können weiterhin mit den erforderlichen KMS-Berechtigungen auf die vorhandenen verschlüsselten Objekte im Datenkatalog zugreifen.

> **Important**
>
> Der AWS KMS Schlüssel muss im AWS KMS Schlüsselspeicher für alle damit verschlüsselten Objekte im Datenkatalog verfügbar bleiben. Wenn Sie den Schlüssel entfernen, können die Objekte nicht mehr entschlüsselt werden. In einigen Szenarien kann dies sinnvoll sein, um den Zugriff auf Metadaten des Data Catalog zu verhindern.

Überwachen Sie Ihre KMS-Schlüssel für AWS Glue

Wenn Sie KMS-Schlüssel mit Ihren Datenkatalogressourcen verwenden, können Sie AWS CloudTrail oder Amazon CloudWatch Logs verwenden, um Anfragen zu verfolgen, die AWS Glue an gesendet

AWS KMS werden. AWS CloudTrail überwacht und zeichnet KMS-Vorgänge auf, AWS Glue die auf Daten zugreifen, die mit Ihren KMS-Schlüsseln verschlüsselt sind.

Bei den folgenden Beispielen handelt es sich um AWS CloudTrail Ereignisse für die `Decrypt` `GenerateDataKey` AND-Operationen.

Decrypt


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAXPHTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-10T14:33:56Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-01-10T15:18:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
```

```
    },
    "responseElements": null,
    "requestID": "43b019aa-34b8-4798-9b98-ee968b2d63df",
    "eventID": "d7614763-d3fe-4f84-a1e1-3ca4d2a5bbd5",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:<region>:111122223333:key/<key-id>"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

GenerateDataKey

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId":
 "AROAXPHTESTANDEXAMPLE:V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/
V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-05T21:15:47Z",
```

```
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-01-05T21:15:47Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:eu-west-2:AKIAIOSFODNN7EXAMPLE:key/
AKIAIOSFODNN7EXAMPLE",
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "keySpec": "AES_256"
    },
    "responseElements": null,
    "requestID": "64d1783a-4b62-44ba-b0ab-388b50188070",
    "eventID": "1c73689b-2ef2-443b-aed7-8c126585ca5e",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-west-2:111122223333:key/AKIAIOSFODNN7EXAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Verschlüsselung von Verbindungspasswörtern

Sie können Verbindungskennwörter in der abrufen, AWS Glue Data Catalog indem Sie die `GetConnections` API-Operationen `GetConnection` und verwenden. Diese Passwörter werden in der Datenkatalogverbindung gespeichert und werden verwendet, wenn AWS Glue stellt eine Verbindung zu einem JDBC-Datenspeicher (Java Database Connectivity) her. Beim Herstellen oder Aktualisieren der Verbindung wurde durch eine Option in den Einstellungen des Datenkatalogs bestimmt, ob das Passwort verschlüsselt war, und falls ja, welcher Schlüssel AWS Key Management Service (AWS KMS) angegeben wurde.

Auf dem AWS Glue In der Konsole können Sie diese Option auf der Seite mit den Einstellungen für den Datenkatalog aktivieren.

Verschlüsseln von Verbindungspasswörtern

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Wählen Sie im Navigationsbereich Settings (Einstellungen) aus.
3. Aktivieren Sie auf der Seite Data catalog settings (Data Catalog-Einstellungen) Encrypt connection passwords (Verbindungspasswörter verschlüsseln) und wählen Sie einen AWS KMS -Schlüssel.

   > **Important**
   >
   > AWS Glue unterstützt nur symmetrische Kunden-Masterschlüssel (CMKs). In der Liste der AWS KMS -Schlüssel werden nur symmetrische Schlüssel angezeigt. Wenn Sie jedoch „ AWS KMS Schlüssel-ARN auswählen“ auswählen, können Sie in der Konsole einen ARN für einen beliebigen Schlüsseltyp eingeben. Stellen Sie sicher, dass Sie nur ARNs für symmetrische Schlüssel eingeben.

   Weitere Informationen finden Sie unter Einstellungen für den Datenkatalog .

Verschlüsseln von Daten, die von AWS Glue geschrieben werden

Eine Sicherheitskonfiguration besteht aus einer Reihe von Sicherheitseigenschaften, die verwendet werden können von AWS Glue. Sie können eine Sicherheitskonfiguration verwenden, um Daten im

Ruhezustand zu verschlüsseln. Die folgenden Szenarien zeigen einige der Möglichkeiten, wie Sie eine Sicherheitskonfiguration verwenden können.

- Hängen Sie eine Sicherheitskonfiguration an eine an AWS Glue Crawler zum Schreiben verschlüsselter CloudWatch Amazon-Logs. Weitere Informationen zum Anhängen von Sicherheitskonfigurationen an Crawler finden Sie unter. the section called “Sicherheitseinstellungen konfigurieren”
- Hängen Sie eine Sicherheitskonfiguration an einen ETL-Job (Extrahieren, Transformieren und Laden) an, um verschlüsselte Amazon Simple Storage Service (Amazon S3) -Ziele und verschlüsselte CloudWatch Protokolle zu schreiben.
- Fügen Sie eine Sicherheitskonfiguration an einen ETL-Auftrag an, um seine Auftragslesezeichen als verschlüsselte Amazon-S3-Daten zu schreiben.
- Fügen Sie eine Sicherheitskonfiguration an einen Entwicklungsendpunkt an, um verschlüsselte Amazon-S3-Ziele zu schreiben.

> **Important**
>
> Derzeit überschreibt eine Sicherheitskonfiguration jede serverseitige Verschlüsselungseinstellung (SSE-S3), die als ETL-Auftragsparameter übergeben wird. Wenn also sowohl eine Sicherheitskonfiguration als auch ein SSE-S3-Parameter einem Auftrag zugeordnet sind, wird der SSE-S3-Parameter ignoriert.

Weitere Informationen zu den Sicherheitskonfigurationen finden Sie unter Sicherheitskonfigurationen auf der AWS Glue Konsole verwalten.

Themen

- Einrichten AWS Glue um Sicherheitskonfigurationen zu verwenden
- Eine Route AWS KMS für VPC-Jobs und Crawler erstellen
- Sicherheitskonfigurationen auf der AWS Glue Konsole verwalten

Einrichten AWS Glue um Sicherheitskonfigurationen zu verwenden

Gehen Sie wie folgt vor, um Ihre einzurichten AWS Glue Umgebung zur Verwendung von Sicherheitskonfigurationen.

1. Erstellen oder aktualisieren Sie Ihre AWS Key Management Service (AWS KMS) -Schlüssel, um den IAM-Rollen, an die übergeben werden, AWS KMS Berechtigungen zu gewähren AWS Glue Crawler und Jobs zum Verschlüsseln CloudWatch von Logs. Weitere Informationen finden Sie unter Verschlüsseln von Protokolldaten in CloudWatch Logs Using AWS KMS im Amazon CloudWatch Logs-Benutzerhandbuch.

   Im folgenden Beispiel *"role3"* sind*"role1"*, und IAM-Rollen*"role2"*, die an Crawler und Jobs übergeben werden.

```
{
       "Effect": "Allow",
       "Principal": { "Service": "logs.region.amazonaws.com",
       "AWS": [
                "role1",
                "role2",
                "role3"
             ] },
                    "Action": [
                           "kms:Encrypt*",
                           "kms:Decrypt*",
                           "kms:ReEncrypt*",
                           "kms:GenerateDataKey*",
                           "kms:Describe*"
                    ],
                    "Resource": "*"
}
```

   Die `Service` Anweisung, dargestellt als, ist erforderlich`"Service": "logs.region.amazonaws.com"`, wenn Sie den Schlüssel zum Verschlüsseln CloudWatch von Protokollen verwenden.

2. Stellen Sie sicher, dass der AWS KMS Schlüssel vorhanden ist, `ENABLED` bevor er verwendet wird.

> **Note**
>
> Wenn Sie Iceberg als Ihr Data Lake Framework verwenden, verfügen die Iceberg-Tabellen über eigene Verfahren, die serverseitige Verschlüsselung ermöglichen. Sie sollten diese Konfiguration zusätzlich zu den AWS Glue Sicherheitskonfigurationen aktivieren. Um die

serverseitige Verschlüsselung für Iceberg-Tabellen zu aktivieren, lesen Sie die Anleitung in der Iceberg-Dokumentation.

Eine Route AWS KMS für VPC-Jobs und Crawler erstellen

Sie können sich direkt mit AWS KMS über einen privaten Endpunkt in Ihrer Virtual Private Cloud (VPC) verbinden, anstatt sich über das Internet zu verbinden. Wenn Sie einen VPC-Endpunkt verwenden, AWS KMS erfolgt die Kommunikation zwischen Ihrer VPC und dem vollständig innerhalb des AWS Netzwerks.

Sie können einen AWS KMS VPC-Endpunkt innerhalb einer VPC erstellen. Ohne diesen Schritt können Ihre Aufträge oder Crawler mit einem `kms timeout` auf Aufträgen oder `internal service exception` auf Crawlern fehlschlagen. Ausführliche Anweisungen finden Sie unter Herstellen einer AWS KMS Verbindung zu einem VPC-Endpunkt im AWS Key Management Service Entwicklerhandbuch.

Wenn Sie diese Anweisungen befolgen, müssen Sie auf der VPC-Konsole Folgendes tun:

- Wählen Sie Enable Private DNS name (Privaten DNS-Namen aktivieren) aus.
- Wählen Sie die Security group (Sicherheitsgruppe) (mit selbstreferenzierender Regel), die Sie für Ihren Auftrag oder Crawler verwenden, der auf die Java Database Connectivity (JDBC) zugreift. Weitere Informationen zur AWS Glue Verbindungen finden Sie unterHerstellen einer Verbindung zu Daten.

Wenn Sie einem Crawler oder Job, der auf JDBC-Datenspeicher zugreift, eine Sicherheitskonfiguration hinzufügen, AWS Glue muss eine Route zum Endpunkt haben. AWS KMS Sie können die Route mit einem NAT-Gateway (Network Address Translation) oder mit einem AWS KMS VPC-Endpunkt bereitstellen. Informationen zum Erstellen eines NAT-Gateways finden Sie unter NAT-Gateways im Amazon-VPC-Benutzerhandbuch.

Sicherheitskonfigurationen auf der AWS Glue Konsole verwalten

Warning

AWS Glue Sicherheitskonfigurationen werden derzeit in Ray-Jobs nicht unterstützt.

Eine Sicherheitskonfiguration in AWS Glue enthält die Eigenschaften, die erforderlich sind, wenn Sie verschlüsselte Daten schreiben. Sie erstellen Sicherheitskonfigurationen auf der AWS Glue-Konsole, um die Verschlüsselungseigenschaften bereitzustellen, die von Crawlern, Aufträgen und Entwicklungsendpunkten verwendet werden.

Zum Anzeigen einer Liste aller Sicherheitskonfigurationen, die Sie erstellt haben, öffnen Sie die AWS Glue-Konsole über https://console.aws.amazon.com/glue/ und wählen Sie im Navigationsbereich Security configurations (Sicherheitskonfigurationen) aus.

Die Liste der Sicherheitskonfigurationen zeigt die folgenden Eigenschaften für jede Konfiguration an:

Name

Der eindeutige Name, den Sie bei der Erstellung der Konfiguration angegeben haben. Der Name darf Buchstaben (A–Z), Zahlen (0–9), Bindestriche (-) oder Unterstriche (_) enthalten und bis zu 255 Zeichen lang sein.

Amazon-S3-Verschlüsselung aktivieren

Wenn diese Option aktiviert ist, wird der Amazon Simple Storage Service (Amazon S3)-Verschlüsselungsmodus wie zum Beispiel `SSE-KMS` oder `SSE-S3` für Metadatenspeicherung im Datenkatalog verwendet.

Verschlüsselung von CloudWatch Amazon-Protokollen aktivieren

Wenn aktiviert, wird der Amazon S3 S3-Verschlüsselungsmodus `SSE-KMS` aktiviert, z. B. beim Schreiben von Protokollen an Amazon CloudWatch.

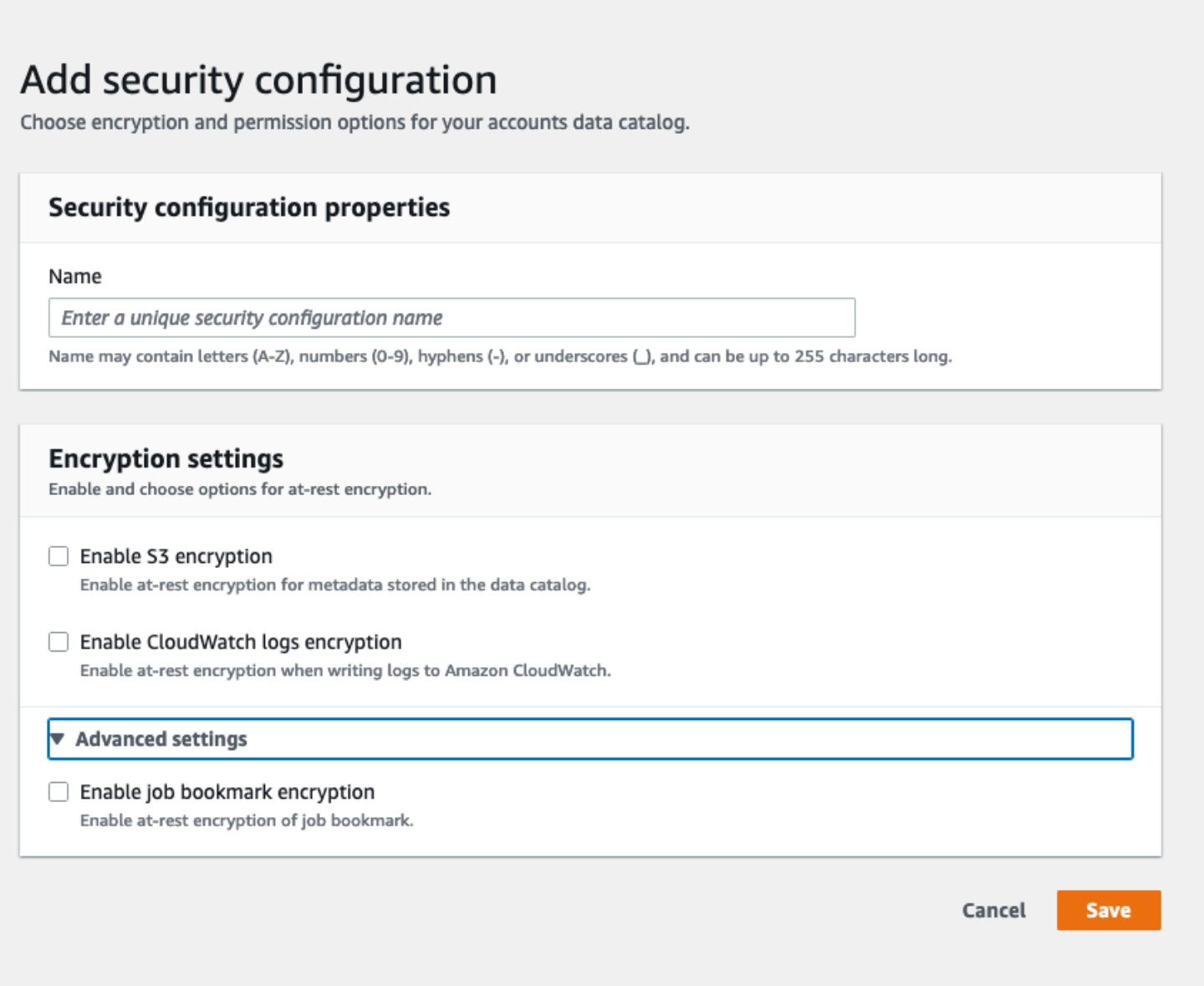
Erweiterte Einstellungen: Verschlüsselung von Auftrags-Lesezeichen aktivieren

Wenn diese Option aktiviert ist, wird der Amazon S3-Verschlüsselungsmodus (z. B. `CSE-KMS`) aktiviert, wenn Aufträge mit Lesezeichen versehen werden.

Sie können Konfigurationen im Abschnitt Security configurations (Sicherheitskonfigurationen) auf der Konsole hinzufügen oder löschen. Um weitere Details über eine Konfiguration zu sehen, wählen Sie den Namen der Konfiguration in der Liste aus. Zu den Details gehören die Informationen, die Sie beim Erstellen der Konfiguration definiert haben.

Hinzufügen einer Sicherheitskonfiguration

Um eine Sicherheitskonfiguration mithilfe der AWS Glue-Konsole hinzuzufügen, wählen Sie auf der Seite Security configurations (Sicherheitskonfigurationen) Add security configuration (Sicherheitskonfigurationen hinzufügen).



Eigenschaften der Sicherheitskonfiguration

Geben Sie einen eindeutigen Namen für die Sicherheitskonfiguration ein. Der Name darf Buchstaben (A–Z), Zahlen (0–9), Bindestriche (-) oder Unterstriche (_) enthalten und bis zu 255 Zeichen lang sein.

Verschlüsselungseinstellungen

Sie können die Verschlüsselung im Ruhezustand für Metadaten aktivieren, die im Datenkatalog in Amazon S3 und in Protokollen in Amazon CloudWatch gespeichert sind. Um die Verschlüsselung von Daten und Metadaten mit AWS Key Management Service (AWS KMS) -Schlüsseln auf der AWS Glue Konsole einzurichten, fügen Sie dem Konsolenbenutzer eine Richtlinie hinzu. Diese Richtlinie

muss die zulässigen Ressourcen als wichtige Amazon-Ressourcennamen (ARNs) angeben, die zur Verschlüsselung von Amazon S3-Datenspeichern verwendet werden, wie im folgenden Beispiel.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt",
            "kms:Encrypt"
        ],
        "Resource": "arn:aws:kms:us-east-1:account-id:key/key-id"
    }
}
```

> **Important**
>
> Wenn eine Sicherheitskonfiguration an einen Crawler oder Job angehängt wird, muss die übergebene IAM-Rolle über Berechtigungen verfügen. AWS KMS Weitere Informationen finden Sie unter Verschlüsseln von Daten, die von AWS Glue geschrieben werden.

Wenn Sie eine Konfiguration definieren, können Sie Werte für die folgenden Eigenschaften angeben:

S3-Verschlüsselung aktivieren

Wenn Sie Amazon S3 S3-Daten schreiben, verwenden Sie entweder serverseitige Verschlüsselung mit verwalteten Amazon S3 S3-Schlüsseln (SSE-S3) oder serverseitige Verschlüsselung mit AWS KMS verwalteten Schlüsseln (SSE-KMS). Dies ist ein optionales Feld. Um den Zugriff auf Amazon S3 zu ermöglichen, wählen Sie einen AWS KMS Schlüssel aus, oder wählen Sie Enter a key ARN und geben Sie den ARN für den Schlüssel ein. Geben Sie den ARN in der Form `arn:aws:kms:region:account-id:key/key-id` ein. Sie können den ARN auch in Form eines Schlüssel-Alias bereitstellen, wie z. B. `arn:aws:kms:region:account-id:alias/alias-name`.

Wenn Sie die Spark-Benutzeroberfläche für Ihren Auftrag aktivieren, wird die auf Amazon S3 hochgeladene Spark-Benutzeroberflächen-Protokolldatei mit derselben Verschlüsselung angewendet.

> **Important**
>
> AWS Glueunterstützt nur symmetrische Kunden-Masterschlüssel (CMKs). In der Liste der AWS KMS -Schlüssel werden nur symmetrische Schlüssel angezeigt. Wenn Sie jedoch „ AWS KMS Schlüssel-ARN auswählen“ auswählen, können Sie in der Konsole einen ARN für einen beliebigen Schlüsseltyp eingeben. Stellen Sie sicher, dass Sie nur ARNs für symmetrische Schlüssel eingeben.

Aktivieren Sie die CloudWatch Protokollverschlüsselung

Die serverseitige Verschlüsselung (SSE-KMS) wird zur Verschlüsselung von Protokollen verwendet. CloudWatch Dies ist ein optionales Feld. Um ihn zu aktivieren, wählen Sie einen AWS KMS Schlüssel aus, oder wählen Sie Einen Schlüssel-ARN eingeben und geben Sie den ARN für den Schlüssel ein. Geben Sie den ARN in der Form `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`* ein. Sie können den ARN auch in Form eines Schlüssel-Alias bereitstellen, wie z. B. `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Erweiterte Einstellungen: Verschlüsselung von Auftrags-Lesezeichen

Es wird eine Client-seitige Verschlüsselung (CSE-KMS) verwendet, um Auftrags-Lesezeichen zu verschlüsseln. Dies ist ein optionales Feld. Die Lesezeichendaten werden verschlüsselt, bevor sie zur Speicherung an Amazon S3 gesendet werden. Um ihn zu aktivieren, wählen Sie einen AWS KMS Schlüssel aus, oder wählen Sie Einen Schlüssel-ARN eingeben und geben Sie den ARN für den Schlüssel ein. Geben Sie den ARN in der Form `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`* ein. Sie können den ARN auch in Form eines Schlüssel-Alias bereitstellen, wie z. B. `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Weitere Informationen finden Sie in den folgenden Themen im Benutzerhandbuch zum Amazon Simple Storage Service:

- Weitere Informationen über `SSE-S3` finden Sie unter Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) (Schutz von Daten durch serverseitige Verschlüsselung mit Amazon S3-Managed Encryption Keys (SSE-S3)).

- Weitere Informationen dazu `SSE-KMS` finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung mit AWS KMS keys.
- Informationen zu `CSE-KMS` finden Sie unter Verwenden eines in AWS KMS gespeicherten KMS-Schlüssels.

Verschlüsseln von Daten während der Übertragung

AWS bietet Transport Layer Security (TLS) -Verschlüsselung für übertragene Daten. Sie können Verschlüsselungseinstellungen für Crawler, ETL-Jobs und Entwicklungsendpunkte mithilfe von Sicherheitskonfigurationen in konfigurieren AWS Glue. Sie können die AWS Glue Data Catalog Verschlüsselung über die Einstellungen für den Datenkatalog aktivieren.

Ab dem 4. September 2018 AWS KMS (bringen Sie Ihren eigenen Schlüssel und serverseitige Verschlüsselung mit) für AWS Glue ETL und das AWS Glue Data Catalog wird unterstützt.

Compliance mit FIPS

Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

Schlüsselverwaltung

Sie können (IAM) verwenden mit AWS Identity and Access Management AWS Glue um Benutzer, AWS Ressourcen, Gruppen, Rollen und detaillierte Richtlinien für Zugriff, Verweigerung und mehr zu definieren.

Sie können den Zugriff auf die Metadaten, je nach den Anforderungen Ihrer Organisation, mit ressourcenbasierten und identitätsbasierten Richtlinien, festlegen. Ressourcenbasierte Richtlinien führen die Prinzipale auf, die für Ihre Ressourcen zugelassen oder verweigert werden, sodass Sie Richtlinien wie z. B. den kontoübergreifenden Zugriff einrichten können. Identitätsrichtlinien sind speziell für Benutzer, Gruppen und Rollen in IAM angefügt.

Ein step-by-step Beispiel finden Sie im Big Data-Blog unter Beschränken Sie den Zugriff auf Ihr System AWS Glue Data Catalog mit IAM-Berechtigungen auf Ressourcenebene und ressourcenbasierten Richtlinien. AWS

Die differenzierte Zugriffskontrolle der Richtlinie ist innerhalb der `Resource`-Klausel definiert. In diesem Abschnitt wird sowohl das AWS Glue Data Catalog Objekt definiert, für das die Aktion ausgeführt werden kann, als auch die resultierenden Objekte, die bei diesem Vorgang zurückgegeben werden.

Ein Entwicklungsendpunkt ist eine Umgebung, die Sie zum Entwickeln und Testen Ihres AWS Glue Skripte. Sie können den SSH-Schlüssel eines Entwicklungsendpunkts hinzufügen, löschen oder rotieren.

Ab dem 4. September 2018 AWS KMS (bringen Sie Ihren eigenen Schlüssel und serverseitige Verschlüsselung mit) für AWS Glue ETL und das AWS Glue Data Catalog wird unterstützt.

AWS Glue Abhängigkeit von anderen AWS Diensten

Damit ein Benutzer mit dem arbeiten kann AWS Glue Dieser Benutzer muss über Mindestberechtigungen verfügen, die es ihm ermöglichen, mit der Konsole zu arbeiten AWS Glue Ressourcen für ihr AWS Konto. Zusätzlich zu diesen AWS Glue Berechtigungen, für die Konsole sind Berechtigungen der folgenden Dienste erforderlich:

- Amazon CloudWatch Logs-Berechtigungen zum Anzeigen von Protokollen.
- AWS Identity and Access Management (IAM) -Berechtigungen zum Auflisten und Weitergeben von Rollen.
- AWS CloudFormation Berechtigungen zum Arbeiten mit Stacks.
- Amazon Elastic Compute Cloud (Amazon EC2) -Berechtigungen zum Auflisten von virtuellen privaten Clouds (VPCs), Subnetzen, Sicherheitsgruppen, Instances und anderen Objekten (zum Einrichten von EC2 Amazon-Elementen, z. B. VPCs beim Ausführen von Jobs, Crawlern und beim Erstellen von Entwicklungsendpunkten).
- Amazon Simple Storage Service (Amazon S3)-Berechtigungen zum Auflisten von Buckets und Objekten sowie zum Abrufen und Speichern von Skripts.
- Amazon-Redshift-Berechtigungen für die Arbeit mit Clustern.
- Amazon Relational Database Service (Amazon RDS)-Berechtigungen zum Auflisten von Instances.

Entwicklungsendpunkte

Ein Entwicklungsendpunkt ist eine Umgebung, in der Sie Ihr System entwickeln und testen können AWS Glue Skripte. Sie können Folgendes verwenden … AWS Glue um Entwicklungsendpunkte zu

erstellen, zu bearbeiten und zu löschen. Sie können alle erstellten Entwicklungsendpunkte auflisten. Sie können den SSH-Schlüssel eines Entwicklungsendpunkts hinzufügen, löschen oder rotieren. Sie können außerdem Notebooks erstellen, die den Entwicklungsendpunkt verwenden.

Sie geben Konfigurationswerte für die Bereitstellung der Entwicklungsumgebungen an. Diese Werte sagen AWS Glue wie Sie das Netzwerk so einrichten, dass Sie sicher auf den Entwicklungsendpunkt zugreifen können und dass Ihr Endpunkt auf Ihre Datenspeicher zugreifen kann. Anschließend erstellen Sie ein Notizbuch (Notebook), das sich mit dem Entwicklungsendpunkt verbindet. Sie verwenden das Notizbuch zum Erstellen und Testen Ihres ETL-Skripts.

Verwenden Sie eine AWS Identity and Access Management (IAM-) Rolle mit ähnlichen Berechtigungen wie die IAM-Rolle, die Sie zum Ausführen verwenden AWS Glue ETL-Jobs. Verwenden Sie eine Virtual Private Cloud (VPC), ein Subnetz und eine Sicherheitsgruppe, um einen Entwicklungsendpunkt zu erstellen, der sich sicher mit Ihren Datenressourcen verbinden kann. Sie generieren ein SSH-Schlüsselpaar, um eine Verbindung mit der Entwicklungsumgebung über SSH her.

Sie können Entwicklungsendpunkte für Amazon-S3-Daten und innerhalb einer VPC erstellen, die Sie verwenden können, um mithilfe von JDBC auf Datensätze zuzugreifen.

Sie können einen Jupyter Notebook-Client auf Ihrem lokalen Computer installieren und ihn zum Debuggen und Testen von ETL-Skripts auf einem Entwicklungsendpunkt verwenden. Oder Sie können ein Sagemaker-Notizbuch verwenden, um ETL-Skripte in JupyterLab ON zu verfassen. AWS Weitere Informationen finden Sie unter Verwenden eines SageMaker Notebooks mit Ihrem Entwicklungsendpunkt.

AWS Glue kennzeichnet EC2 Amazon-Instances mit einem Namen, dem ein Präfix vorangestellt ist. `aws-glue-dev-endpoint`

Sie können einen Notebook-Server auf einem Entwicklungsendpunkt einrichten, mit dem er ausgeführt werden PySpark kann AWS Glue Erweiterungen.

Identitäts- und Zugriffsmanagement für AWS Glue

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu AWS kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um AWS Glue-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

> **Note**
>
> Sie können den Zugriff auf Ihre Daten im AWS Glue-Datenkatalog entweder mit AWS Glue Methoden oder mit AWS Lake Formation Zuschüssen gewähren. Sie können AWS Identity and Access Management (IAM) -Richtlinien verwenden, um eine differenzierte Zugriffskontrolle mit Methoden festzulegen. AWS Glue Lake Formation verwendet ein einfacheres `GRANT/REVOKE`-Berechtigungsmodell ähnlich den `GRANT/REVOKE`-Befehlen in einem relationalen Datenbanksystem.
> Dieser Abschnitt enthält Informationen dazu, wie Sie diese AWS Glue-Methoden verwenden. Weitere Informationen zur Verwendung von Lake-Formation-Berechtigungen finden Sie unter Erteilen von Lake-Formation-Berechtigungen im AWS Lake Formation -Entwicklerhandbuch.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So funktioniert AWS Glue mit IAM
- Konfiguration von IAM-Berechtigungen für AWS Glue
- AWS Beispiele für Richtlinien zur Glue-Zugriffskontrolle
- Gewährung AWS verwalteter Richtlinien für AWS Glue
- Erteilen von Richtlinien mit dynamischem Geltungsbereich für die Auftragsausführung
- AWS GlueRessource angeben ARNs
- Gewährung von kontenübergreifendem Zugriff
- Fehlerbehebung bei Identität und Zugriff auf AWS Glue

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in AWS Glue ausführen.

Dienstbenutzer — Wenn Sie den AWS Glue-Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen zur Verfügung, die Sie benötigen. Da Sie für Ihre Arbeit mehr Funktionen von AWS Glue verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen,

wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Unter Fehlerbehebung bei Identität und Zugriff auf AWS Glue finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf ein Feature in AWS Glue haben.

Service-Administrator — Wenn Sie in Ihrem Unternehmen für die AWS Glue-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf AWS Glue. Es ist Ihre Aufgabe, zu bestimmen, auf welche Funktionen und Ressourcen von AWS Glue Ihre Servicebenutzer zugreifen sollen. Anschließend müssen Sie Anforderungen an Ihren IAM-Administrator senden, um die Berechtigungen der Servicebenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM mit AWS Glue verwenden kann, finden Sie unterSo funktioniert AWS Glue mit IAM.

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien schreiben können, um den Zugriff auf Glue zu verwalten. AWS Beispiele für identitätsbasierte Richtlinien von AWS Glue, die Sie in IAM verwenden können, finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Glue AWS

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, mit denen Sie Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch signieren können. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen

Methode für die Selbstsignierung von Anforderungen finden Sie unter AWS Signature Version 4 für API-Anforderungen im IAM-Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen bereitstellen. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und AWS Multi-Faktor-Authentifizierung (MFA) in IAM im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen. Verwenden Sie diese nur, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise eine Gruppe benennen IAMAdminsund dieser Gruppe Berechtigungen zur Verwaltung von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb von Ihnen AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, jedoch nicht mit einer bestimmten Person verknüpft. Um vorübergehend eine IAM-Rolle in der zu übernehmen AWS Management Console, können Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Methoden für die Übernahme einer Rolle im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe

Identitätsanbieter (Verbund) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise in einem Service einen Anruf tätigen, ist es üblich, dass dieser Service Anwendungen in Amazon ausführt EC2 oder Objekte in Amazon S3 speichert. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
  - Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und

gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

- Auf Amazon ausgeführte Anwendungen EC2 — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2 Instance ausgeführt werden und AWS API-Anfragen stellen AWS CLI . Dies ist dem Speichern von Zugriffsschlüsseln innerhalb der EC2 Instance vorzuziehen. Um einer EC2 Instanz eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instanzprofil, das an die Instanz angehängt ist. Ein Instanzprofil enthält die Rolle und ermöglicht Programmen, die auf der EC2 Instanz ausgeführt werden, temporäre Anmeldeinformationen abzurufen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Verwenden einer IAM-Rolle, um Berechtigungen für Anwendungen zu gewähren, die auf EC2 Amazon-Instances ausgeführt werden.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Die Berechtigungen in den Richtlinien legen fest, ob eine Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer Inline-Richtlinie wählen, finden Sie unter Auswählen zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffskontrolllisten () ACLs

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Dienste, die Unterstützung ACLs bieten. AWS WAF Weitere Informationen finden Sie unter Übersicht über ACLs die Zugriffskontrollliste (ACL) im Amazon Simple Storage Service Developer Guide.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Dienststeuerungsrichtlinien (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie alle Funktionen in einer Organisation aktivieren, können Sie Richtlinien zur Servicesteuerung (SCPs) auf einige oder alle Ihre Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Root-Benutzer des AWS-Kontos Entitäten. Weitere Informationen zu Organizations und SCPs finden Sie unter Richtlinien zur Servicesteuerung im AWS Organizations Benutzerhandbuch.
- Ressourcenkontrollrichtlinien (RCPs) — RCPs sind JSON-Richtlinien, mit denen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten festlegen können, ohne die IAM-Richtlinien aktualisieren zu müssen, die jeder Ressource zugeordnet sind, deren Eigentümer Sie sind. Das RCP schränkt die Berechtigungen für Ressourcen in Mitgliedskonten ein und kann sich auf die effektiven Berechtigungen für Identitäten auswirken, einschließlich der Root-Benutzer des AWS-Kontos, unabhängig davon, ob sie zu Ihrer Organisation gehören. Weitere Informationen zu Organizations RCPs, einschließlich einer Liste AWS-Services dieser Support-Leistungen RCPs, finden Sie unter Resource Control Policies (RCPs) im AWS Organizations Benutzerhandbuch.

- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter Bewertungslogik für Richtlinien.

So funktioniert AWS Glue mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf AWS Glue zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen mit AWS Glue verwendet werden können.

IAM-Funktionen, die Sie mit AWS Glue verwenden können

IAM-Feature	AWS Unterstützung für Glue
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Teilweise
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Richtlinienbedingungsschlüssel (services pezifisch)	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise

IAM-Feature	AWS Unterstützung für Glue
Temporäre Anmeldeinformationen	Ja
Hauptberechtigungen	Nein
Servicerollen	Ja
Service-verknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie AWS Glue und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für Glue AWS

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

AWS Glue unterstützt identitätsbasierte Richtlinien (IAM-Richtlinien) für alle AWS Glue Operationen. Durch Anhängen einer Richtlinie können Sie Berechtigungen zum Erstellen, Zugreifen oder Ändern eines AWS Glue Ressource, z. B. eine Tabelle in der AWS Glue Data Catalog.

Beispiele für identitätsbasierte Richtlinien für Glue AWS

Beispiele für identitätsbasierte Richtlinien von AWS Glue finden Sie unter. Beispiele für identitätsbasierte Richtlinien für Glue AWS

Ressourcenbasierte Richtlinien innerhalb von Glue AWS

Unterstützt ressourcenbasierte Richtlinien: Teilweise

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM-Administrator des vertrauenswürdigen Kontos auch der Prinzipalentität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

> **Note**
>
> Sie können nur eine verwenden AWS Glue Ressourcenrichtlinie zur Verwaltung von Berechtigungen für Datenkatalogressourcen. Sie können es nicht an andere anhängen AWS Glue Ressourcen wie Jobs, Trigger, Entwicklungsendpunkte, Crawler oder Classifier. Derzeit ist pro Katalog nur eine Ressourcenrichtlinie zulässig, deren Größe auf 10 KB begrenzt ist.

In AWS Glue ist eine Ressourcenrichtlinie an einen Katalog angehängt, der ein virtueller Container für alle zuvor genannten Arten von Datenkatalogressourcen ist. Jedes AWS Konto besitzt einen

einzelnen Katalog in einer AWS Region, dessen Katalog-ID mit der AWS Konto-ID identisch ist. Sie können einen Katalog nicht löschen oder ändern.

Eine Ressourcenrichtlinie wird für alle API-Aufrufe an den Katalog ausgewertet, bei denen das Aufruferprinzip in den `"Principal"`-Block des Richtliniendokuments eingebunden ist.

Beispiele für ressourcenbasierte Richtlinien von AWS Glue finden Sie unter. Beispiele für ressourcenbasierte Richtlinien für Glue AWS

Politische Maßnahmen für AWS Glue

Unterstützt Richtlinienaktionen: Ja

Administratoren können AWS JSON-Richtlinien verwenden, um festzulegen, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der AWS Glue-Aktionen finden Sie unter Von AWS Glue definierte Aktionen in der Serviceautorisierungsreferenz.

Richtlinienaktionen in AWS Glue verwenden vor der Aktion das folgende Präfix:

```
glue
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
```

```
      "glue:action1",
      "glue:action2"
         ]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `Get` beginnen, einschließlich der folgenden Aktion:

```
"Action": "glue:Get*"
```

Informationen zum Anzeigen von Beispielrichtlinien finden Sie unter AWS Beispiele für Richtlinien zur Glue-Zugriffskontrolle.

Politische Ressourcen für AWS Glue

Unterstützt Richtlinienressourcen: Ja

Administratoren können AWS JSON-Richtlinien verwenden, um festzulegen, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein – `Resource`oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Weitere Informationen zur Steuerung des Zugriffs auf AWS Glue-Ressourcen mithilfe von ARNs finden Sie unterAWS GlueRessource angeben ARNs.

Eine Liste der AWS Glue-Ressourcentypen und ihrer ARNs Eigenschaften finden Sie unter Von AWS Glue definierte Ressourcen in der Service Authorization Reference. Informationen darüber, mit

welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von AWS Glue definierte Aktionen.

Schlüssel zu den Policy-Bedingungen für AWS Glue

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, AWS wertet die Bedingung mithilfe einer logischen `OR` Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der AWS Glue-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für AWS Glue in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von AWS Glue definierte Aktionen.

Informationen zum Anzeigen von Beispielrichtlinien finden Sie unter Einstellungen über Bedingungsschlüssel oder Kontextschlüssel steuern.

ACLs in AWS Glue

Unterstützt ACLs: Nein

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit Glue AWS

Unterstützt ABAC (Tags in Richtlinien): Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In AWS werden diese Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

> **Important**
>
> Die Bedingungskontextschlüssel gelten nur für AWS Glue API-Aktionen für Crawler, Jobs, Trigger und Entwicklungsendpunkte. Weitere Informationen darüber, welche API-Operationen betroffen sind, finden Sie unter Bedingungsschlüssel für AWS Glue.

> Das Tool AWS Glue Datenkatalog-API-Operationen unterstützen derzeit die Kontextschlüssel `aws:referer` und die `aws:UserAgent` globalen Bedingungsschlüssel nicht.

Ein Beispiel für eine identitätsbasierte Richtlinie zur Einschränkung des Zugriffs auf eine Ressource auf der Grundlage der Markierungen dieser Ressource finden Sie unter Zugriffssteuerung gewähren mit Tags.

Temporäre Anmeldeinformationen mit AWS Glue verwenden

Unterstützt temporäre Anmeldeinformationen: Ja

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services funktionieren AWS-Services , finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM.

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter Wechseln von einer Benutzerrolle zu einer IAM-Rolle (Konsole) im IAM-Benutzerhandbuch.

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Sicherheitsanmeldeinformationen in IAM.

Serviceübergreifende Prinzipalberechtigungen für Glue AWS

Unterstützt Forward Access Sessions (FAS): Nein

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen in auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-

Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Forward Access Sessions (FAS).

Servicerollen für AWS Glue

Unterstützt Servicerollen: Ja

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle kann die AWS Glue-Funktionalität beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn AWS Glue Sie dazu anleitet.

Eine ausführliche Anleitung zum Erstellen einer Servicerolle für AWS Glue finden Sie unter Schritt 1: Erstellen Sie eine IAM-Richtlinie für den AWS Glue-Service undSchritt 2: Erstellen einer IAM-Rolle für AWS Glue.

Servicebezogene Rollen für Glue AWS

Unterstützt serviceverknüpfte Rollen: Ja

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

Konfiguration von IAM-Berechtigungen für AWS Glue

Sie verwenden AWS Identity and Access Management (IAM), um Richtlinien und Rollen zu definieren, die AWS Glue verwendet, um auf Ressourcen zuzugreifen. Die folgenden Schritte führen Sie durch verschiedene Optionen zum Einrichten der Berechtigungen für AWS Glue. Je nach Ihren geschäftlichen Anforderungen können Sie den Zugriff auf Ihre Ressourcen erweitern oder reduzieren.

> **Note**
>
> Informationen zu den ersten Schritten mit grundlegenden IAM-Berechtigungen für AWS Glue instead finden Sie unterEinrichten von IAM-Berechtigungen für AWS Glue.

1. Erstellen Sie eine IAM-Richtlinie für AWS Glue service: Erstellen Sie eine Servicerichtlinie, die den Zugriff auf ermöglicht AWS Glue Ressourcen schätzen.
2. Erstellen Sie eine IAM-Rolle für AWS Glue: Erstellen Sie eine IAM-Rolle und fügen Sie die hinzu AWS Glue Servicerichtlinie und eine Richtlinie für Ihre Amazon Simple Storage Service (Amazon S3) -Ressourcen, die verwendet werden von AWS Glue.
3. Ordnen Sie Benutzern oder Gruppen, die Zugriff darauf haben, eine Richtlinie zu AWS Glue: Hängen Sie Richtlinien an alle Benutzer oder Gruppen an, die sich bei dem anmelden AWS Glue console.
4. Erstellen einer IAM-Richtlinie für Notebooks: Erstellen Sie eine Notebook-Server-Richtlinie für die Erstellung von Notebook-Servern auf Entwicklungsendpunkten.
5. Erstellen einer IAM-Rolle für Notebooks: Erstellen Sie eine IAM-Rolle und fügen Sie die Notebook-Server-Richtlinie an.
6. Erstellen Sie eine IAM-Richtlinie für Amazon SageMaker AI-Notebooks: Erstellen Sie eine IAM-Richtlinie, die bei der Erstellung von Amazon SageMaker AI-Notebooks auf Entwicklungsendpunkten verwendet werden soll.
7. Erstellen Sie eine IAM-Rolle für Amazon SageMaker AI-Notebooks: Erstellen Sie eine IAM-Rolle und fügen Sie die Richtlinie zur Erteilung von Berechtigungen bei der Erstellung von Amazon SageMaker AI-Notebooks auf Entwicklungsendpunkten hinzu.

Schritt 1: Erstellen Sie eine IAM-Richtlinie für den AWS Glue-Service

Für jeden Vorgang, der auf Daten auf einer anderen AWS Ressource zugreift, z. B. für den Zugriff auf Ihre Objekte in Amazon S3, AWS Glue benötigen Sie die Erlaubnis, in Ihrem Namen auf die

Ressource zuzugreifen. Sie gewähren diese Berechtigungen mithilfe von AWS Identity and Access Management (IAM).

> **Note**
>
> Sie können diesen Schritt überspringen, wenn Sie die AWS verwaltete Richtlinie **`AWSGlueServiceRole`** verwenden.

In diesem Schritt erstellen Sie eine ähnliche Richtlinie wie `AWSGlueServiceRole`. Die aktuelle Version von `AWSGlueServiceRole` finden Sie in der IAM-Konsole.

Um eine IAM-Richtlinie zu erstellen für AWS Glue

Diese Richtlinie erteilt die Berechtigung für einige Amazon-S3-Aktionen zur Verwaltung von Ressourcen in Ihrem Konto, die der AWS Glue-Service benötigt, wenn er die Rolle mit dieser Richtlinie annimmt. Einige der in dieser Richtlinie angegebenen Ressourcen beziehen sich auf Standardnamen, die von AWS Glue Amazon S3-Buckets, Amazon S3-ETL-Skripts, CloudWatch Protokollen und EC2 Amazon-Ressourcen verwendet werden. Der Einfachheit halber schreibt AWS Glue einige Amazon-S3-Objekte in Buckets in Ihrem Konto mit dem Standardpräfix `aws-glue-*`.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Navigieren Sie auf dem Bildschirm Create Policy (Richtlinie erstellen) zu einer Registerkarte, um JSON zu bearbeiten. Erstellen Sie ein Richtliniendokument mit den folgenden JSON-Anweisungen und wählen Sie dann Review Policy (Richtlinie überprüfen) aus.

   > **Note**
   >
   > Fügen Sie alle erforderlichen Berechtigungen für Amazon-S3-Ressourcen hinzu. Sie können den Umfang des Ressourcenabschnitts in Ihrer Zugriffsrichtlinie auf die Ressourcen beschränken, die erforderlich sind.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:GetBucketAcl",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeRouteTables",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcAttribute",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "cloudwatch:PutMetricData"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPublicAccessBlock"
            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*/*",
                "arn:aws:s3:::*/*aws-glue-*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::crawler-public*",
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:AssociateKmsKey"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws-glue/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:DeleteTags"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "aws-glue-service-resource"
                    ]
```

```
                }
            },
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:instance/*"
            ]
        }
    ]
}
```

Die folgende Tabelle beschreibt die Berechtigungen, die von dieser Richtlinie erteilt werden.

Action (Aktion)	Ressource	Beschreibung
`"glue:*"`	`"*"`	Gewährt die Berechtigung zum Ausführen aller AWS Glue-API-Operationen.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl",`	`"*"`	Ermöglicht die Auflistung von Amazon-S3-Buckets aus Crawlern, Aufträgen, Entwicklungsendpunkten und Notebook-Servern.
`"ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:CreateNetworkInterface", "ec2:DeleteNetworkInterface", "ec2:DescribeNetworkInterfaces", "ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcAttribute",`	`"*"`	Ermöglicht die Einrichtung von EC2 Amazon-Netzwerkelementen wie virtuellen privaten Clouds (VPCs) beim Ausführen von Jobs, Crawlern und Entwicklungsendpunkten.

Action (Aktion)	Ressource	Beschreibung
`"iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy"`	`"*"`	Ermöglicht die Auflistung von IAM-Rollen aus Crawlern, Aufträgen, Entwicklungsendpunkten und Notebook-Servern.
`"cloudwatch:PutMetricData"`	`"*"`	Ermöglicht das Schreiben von CloudWatch Metriken für Jobs.
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3:::aws-glue-*"`	Ermöglicht das Erstellen von Amazon-S3-Buckets in Ihrem Konto aus Aufträgen und Notebook-Servern. Namenskonvention: Verwendet Amazon-S3-Ordner mit dem Namen aws-glue-. Ermöglicht AWS Glue die Erstellung von Buckets, die den öffentlichen Zugriff blockieren.

Action (Aktion)	Ressource	Beschreibung
`"s3:GetObject", "s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue-*/*", "arn:aws:s3:::*/*aws-glue-*/*"`	Erlaubt das Abrufen, Speichern und Löschen von Amazon-S3-Objekten in Ihrem Konto beim Speichern von Objekten wie ETL-Skripts und Notebook-Server-Standorten. Namenskonvention: Erteilt Berechtigungen für Amazon-S3-Buckets oder Ordner, deren Namen das Präfix aws-glue- enthalten.
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Lässt das Abrufen von Amazon-S3-Objekten, die von Beispielen und Tutorials verwendet werden, aus Crawlern und Aufträgen zu. Namenskonvention: Amazon-S3-Bucket-Namen beginnen mit crawler-public und aws-glue-.
`"logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"`	`"arn:aws:logs:*:*:log-group:/aws-glue/*"`	Ermöglicht das Schreiben von Protokollen in CloudWatch Logs. Namenskonvention: AWS Glue schreibt Protokolle in Protokollgruppen, deren Namen mit aws-glue beginnen.

Action (Aktion)	Ressource	Beschreibung
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Ermöglicht das Taggen von EC2 Amazon-Ressourcen, die für Entwicklungsendpunkte erstellt wurden. Namenskonvention: AWS Glue kennzeichnet EC2 Amazon-Netzwerkschnittstellen, Sicherheitsgruppen und Instances mit aws-glue-service-resource.

5. Geben Sie auf Bildschirm Review Policy (Richtlinie überprüfen) Ihren Policy Name (Richtlinienname) ein, z. B. GlueServiceRolePolicy. Geben Sie eine optionale Beschreibung ein und wählen Sie Create policy (Richtlinie erstellen) aus, wenn Sie mit der Richtlinie zufrieden sind.

Schritt 2: Erstellen einer IAM-Rolle für AWS Glue

Sie müssen Ihrer IAM-Rolle Berechtigungen erteilen, die AWS Glue beim Aufrufen von anderen Services in Ihrem Namen annehmen kann. Dies umfasst den Zugriff auf Amazon S3 für alle Quellen, Ziele, Skripts und temporären Verzeichnisse, die Sie mit AWS Glue verwenden. Die Berechtigung wird von Crawlern, Aufträgen Entwicklungsendpunkten benötigt.

Sie stellen diese Berechtigungen mithilfe von AWS Identity and Access Management (IAM) bereit. Fügen Sie der IAM-Rolle, die Sie an AWS Glue übergeben, eine Richtlinie hinzu.

So erstellen Sie eine IAM-Rolle für AWS Glue

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Wählen Sie Rolle erstellen aus.
4. Wählen Sie AWS Service als vertrauenswürdigen Entitätstyp aus. Suchen Sie dann nach einem Dienst oder einem Anwendungsfall und wählen Sie ihn aus AWS Glue. Wählen Sie Weiter aus.
5. Wählen Sie auf der Seite Berechtigungen hinzufügen die Richtlinien aus, die die erforderlichen Berechtigungen enthalten, z. B. die AWS verwaltete Richtlinie `AWSGlueServiceRole` für

allgemeine AWS Glue Berechtigungen und die AWS verwaltete Richtlinie AmazonS3 FullAccess für den Zugriff auf Amazon S3-Ressourcen. Wählen Sie anschließend Weiter.

> **Note**
>
> Stellen Sie sicher, dass eine der Richtlinien in dieser Rolle Berechtigungen für Ihre Amazon-S3-Quellen und -Ziele erteilt. Sie können Ihre eigenen Richtlinien für den Zugriff auf bestimmte Amazon-S3-Ressourcen bereitstellen. Datenquellen erfordern die Berechtigungen `s3:ListBucket` und `s3:GetObject`. Für Datenziele sind die Berechtigungen `s3:ListBucket`, `s3:PutObject` und `s3:DeleteObject` erforderlich. Weitere Informationen zum Erstellen einer Amazon-S3-Richtlinie für Ihre Ressourcen finden Sie unter Festlegen von Ressourcen in einer Richtlinie. Eine Amazon-S3-Beispielrichtlinie finden Sie unter Schreiben von IAM-Richtlinien: So gewähren Sie Zugriff auf einen Amazon-S3-Bucket.
>
> Wenn Sie vorhaben, auf Amazon-S3-Quellen und -Ziele zuzugreifen, die mit SSE-KMS verschlüsselt sind, fügen Sie eine Richtlinie an, die AWS Glue-Crawlern, Aufträgen und Entwicklungsendpunkten erlaubt, die Daten zu entschlüsseln. Weitere Informationen finden Sie unter Schutz von Daten mithilfe serverseitiger Verschlüsselung mit AWS KMS verwalteten Schlüsseln (SSE-KMS).
>
> Im Folgenden wird ein Beispiel gezeigt.
>
> JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:*:111122223333:key/key-id"
            ]
        }
    ]
}
```

6. Geben Sie Ihrer Rolle einen Namen und fügen Sie eine Beschreibung hinzu (optional). Überprüfen Sie anschließend die Vertrauensrichtlinie und die Berechtigungen. Geben Sie in Role name (Rollenname) einen Namen für die Rolle ein, z. B. `AWSGlueServiceRoleDefault`. Erstellen Sie die Rolle mit dem Namenspräfix mit dem String `AWSGlueServiceRole`, damit die Rolle von Konsolenbenutzern an den Service übergeben werden kann. Die von AWS Glue bereitgestellten Richtlinien erwarten, dass IAM-Servicerollen mit `AWSGlueServiceRole` beginnen. Andernfalls müssen Sie eine Richtlinie hinzufügen, um Ihren Benutzern die `iam:PassRole`-Berechtigung zu erteilen, das IAM-Rollen Ihrer Benennungskonvention entsprechen können. Wählen Sie Rolle erstellen aus.

   > **Note**
   >
   > Wenn Sie ein Notebook mit einer Rolle erstellen, wird diese Rolle dann an interaktive Sitzungen übergeben, sodass dieselbe Rolle an beiden Stellen verwendet werden kann. Daher muss die `iam:PassRole`-Berechtigung Teil der Richtlinie der Rolle sein. Erstellen Sie anhand des folgenden Beispiels eine neue Richtlinie für Ihre Rolle. Ersetzen Sie die Kontonummer durch Ihre eigene und den Rollennamen.
   >
   > JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "iam:PassRole",
               "Resource": "arn:aws:iam::090000000210:role/<role_name>"
           }
       ]
   }
   ```

7. Fügen Sie Ihrer Rolle Tags hinzu (optional). Tags sind Schlüssel-Wert-Paare, die Sie Ressourcen hinzufügen können, um AWS Ressourcen zu identifizieren, zu organisieren oder nach ihnen zu suchen. Wählen Sie anschließend Create role (Rolle erstellen) aus.

Schritt 3: Fügen Sie eine Richtlinie an Benutzer an, die auf AWS Glue zugreifen

Der Administrator muss allen Benutzern, Gruppen oder Rollen mithilfe der AWS Glue Konsole oder AWS Command Line Interface (AWS CLI) Berechtigungen zuweisen. Sie stellen diese Berechtigungen bereit, indem Sie AWS Identity and Access Management (IAM) durch Richtlinien verwenden. Dieser Schritt beschreibt die Zuweisung von Berechtigungen an Benutzer oder Gruppen.

Wenn Sie diesen Schritt abgeschlossen haben, sind Ihrem Benutzer oder Ihrer Gruppe die folgenden Richtlinien angefügt:

- Die AWS verwaltete Richtlinie `AWSGlueConsoleFullAccess` oder die benutzerdefinierte Richtlinie GlueConsoleAccessPolicy
- **`AWSGlueConsoleSageMakerNotebookFullAccess`**
- **`CloudWatchLogsReadOnlyAccess`**
- **`AWSCloudFormationReadOnlyAccess`**
- **`AmazonAthenaFullAccess`**

So fügen Sie eine Inlinerichtlinie an und betten sie in einen Benutzer oder eine Gruppe ein

Sie können einem Benutzer oder einer Gruppe eine AWS verwaltete Richtlinie oder eine Inline-Richtlinie zuordnen, um auf die AWS Glue Konsole zuzugreifen. Einige der in dieser Richtlinie angegebenen Ressourcen beziehen sich auf Standardnamen, die von AWS Glue Amazon S3-Buckets, Amazon S3-ETL-Skripts AWS CloudFormation, CloudWatch Protokollen und EC2 Amazon-Ressourcen verwendet werden. Der Einfachheit halber schreibt AWS Glue einige Amazon-S3-Objekte in Buckets in Ihrem Konto mit dem Standardpräfix `aws-glue-*`.

> **Note**
>
> Sie können diesen Schritt überspringen, wenn Sie die AWS verwaltete Richtlinie **`AWSGlueConsoleFullAccess`**verwenden.

> **Important**
>
> AWS Glue benötigt die Berechtigung zum Annehmen einer Rolle, die verwendet wird, um in Ihrem Namen Aufgaben zu erledigen. Zu diesem Zweck fügen Sie Ihren AWS Glue-Benutzern oder -Gruppen die **`iam:PassRole`**-Berechtigungen hinzu. Diese Richtlinie

erteilt eine Berechtigung für Rollen, die mit `AWSGlueServiceRole` für AWS Glue-Servicerollen beginnen und `AWSGlueServiceNotebookRole` für Rollen, die beim Erstellen eines Notebook-Servers erforderlich sind. Sie können auch eine eigene Richtlinie für `iam:PassRole`-Berechtigungen erstellen, die Ihrer Benennungskonvention entsprechen. Gemäß den bewährten Sicherheitsmethoden wird empfohlen, den Zugriff einzuschränken, indem die Richtlinien verschärft werden, um den Zugriff auf Amazon S3 S3-Buckets und Amazon CloudWatch Protokollgruppen weiter einzuschränken. Eine Amazon-S3-Beispielrichtlinie finden Sie unter Schreiben von IAM-Richtlinien: So gewähren Sie Zugriff auf einen Amazon-S3-Bucket.

In diesem Schritt erstellen Sie eine ähnliche Richtlinie wie `AWSGlueConsoleFullAccess`. Die aktuelle Version von `AWSGlueConsoleFullAccess` finden Sie in der IAM-Konsole.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im Navigationsbereich Benutzer oder Benutzergruppen.
3. Wählen Sie in der Liste den Namen des Benutzers oder der Gruppe, in die eine Richtlinie eingebettet werden soll.
4. Wählen Sie die Registerkarte Permissions (Berechtigungen) aus und erweitern Sie ggf. den Abschnitt Permissions policies (Berechtigungsrichtlinien).
5. Wählen Sie den Link Add Inline policy (Inlinerichtlinie hinzufügen) aus.
6. Navigieren Sie auf dem Bildschirm Create Policy (Richtlinie erstellen) zu einer Registerkarte, um JSON zu bearbeiten. Erstellen Sie ein Richtliniendokument mit den folgenden JSON-Anweisungen und wählen Sie dann Review Policy (Richtlinie überprüfen) aus.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "redshift:DescribeClusters",
                "redshift:DescribeClusterSubnetGroups",
```

```
                "iam:ListRoles",
                "iam:ListUsers",
                "iam:ListGroups",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeRouteTables",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeInstances",
                "rds:DescribeDBInstances",
                "rds:DescribeDBClusters",
                "rds:DescribeDBSubnetGroups",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "cloudformation:DescribeStacks",
                "cloudformation:GetTemplateSummary",
                "dynamodb:ListTables",
                "kms:ListAliases",
                "kms:DescribeKey",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::*/*aws-glue-*/*",
                "arn:aws:s3:::aws-glue-*"
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPublicAccessBlock"            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:GetLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:/aws-glue/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack"
            ],
            "Resource": "arn:aws:cloudformation:*:*:stack/aws-glue*/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:key-pair/*",
```

```
                "arn:aws:ec2:*:*:image/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:volume/*"
            ]
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceNotebookRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"
            ],
            "Condition": {
```

```
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        ]
}
```

Die folgende Tabelle beschreibt die Berechtigungen, die von dieser Richtlinie erteilt werden.

Action (Aktion)	Ressource	Beschreibung
`"glue:*"`	`"*"`	Gewährt die Berechtigung zum Ausführen aller AWS Glue-API-Operationen. Wenn Sie zuvor Ihre Richtlinien ohne die `"glue:*"`-Aktion erstellt haben, müssen Sie die folgenden einzelnen Berechtigungen zu Ihrer Richtlinie hinzufügen: • „Klebstoff:ListCrawlers“ • „kleben:BatchGetCrawlers“ • „kleben:ListTriggers“ • „kleben:BatchGetTriggers“ • „kleben:ListDevEndpoints“ • „kleben:BatchGetDevEndpoints“ • „kleben:ListJobs“ • „kleben:BatchGetJobs“
`"redshift:DescribeClusters", "redshift:DescribeClusterSubnetGroups"`	`"*"`	Ermöglicht das Erstellen von Verbindungen mit Amazon Redshift.

Action (Aktion)	Ressource	Beschreibung
`"iam:ListRoles", "iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy", "iam:ListAttachedRolePolicies"`	`"*"`	Ermöglicht die Auflistung von IAM-Rollen beim Arbeiten mit Crawlern, Aufträgen, Entwicklungsendpunkten und Notebook-Servern.
`"ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcs", "ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:DescribeVpcAttribute", "ec2:DescribeKeyPairs", "ec2:DescribeInstances"`	`"*"`	Ermöglicht die Einrichtung von EC2 Amazon-Netzwerkelementen, z. B. VPCs beim Ausführen von Jobs, Crawlern und Entwicklungsendpunkten.
`"rds:DescribeDBInstances"`	`"*"`	Ermöglicht das Erstellen von Verbindungen mit Amazon RDS.
`"s3:ListAllMyBuckets", "s3:ListBucket", "s3:GetBucketAcl", "s3:GetBucketLocation"`	`"*"`	Ermöglicht die Auflistung von Amazon-S3-Buckets beim Arbeiten mit Crawlern, Aufträgen, Entwicklungsendpunkten und Notebook-Servern.
`"dynamodb:ListTables"`	`"*"`	Ermöglicht die Auflistung von DynamoDB-Tabellen.
`"kms:ListAliases", "kms:DescribeKey"`	`"*"`	Ermöglicht die Arbeit mit KMS-Schlüsseln.

Action (Aktion)	Ressource	Beschreibung
`"cloudwatch:GetMetricData", "cloudwatch:ListDashboards"`	`"*"`	Ermöglicht das Arbeiten mit Metriken CloudWatch .
`"s3:GetObject", "s3:PutObject"`	`"arn:aws:s3::: aws-glue-*/*", "arn:aws:s3::: */*aws-glue-*/*", "arn:aws:s3::: aws-glue-*"`	Erlaubt das Abrufen und Speichern von Amazon-S3-Objekten in Ihrem Konto beim Speichern von Objekten wie ETL-Skripts und Notebook-Server-Standorten. Namenskonvention: Erteilt Berechtigungen für Amazon-S3-Buckets oder Ordner, deren Namen das Präfix aws-glue- enthalten.
`"tag:GetResources"`	`"*"`	Ermöglicht das Abrufen von AWS Tags.

Action (Aktion)	Ressource	Beschreibung
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3::: aws-glue-*"`	Erlaubt das Erstellen eines Amazon-S3-Buckets in Ihrem Konto beim Speichern von Objekten wie ETL-Skripts und Notebook-Server-Standorten. Namenskonvention: Erteilt Berechtigungen für Amazon-S3-Buckets, deren Namen das Präfix aws-glue- enthalten. Ermöglicht AWS Glue die Erstellung von Buckets, die den öffentlichen Zugriff blockieren.
`"logs:GetLogEvents"`	`"arn:aws:logs:*:*: / aws-glue/*"`	Ermöglicht das Abrufen von Protokollen. CloudWatch Namenskonvention: AWS Glue schreibt Protokolle in Protokoll gruppen, deren Namen mit aws-glue- beginnen.

Action (Aktion)	Ressource	Beschreibung
`"cloudformation:CreateStack", "cloudformation:DeleteStack"`	`"arn:aws:cloudformation:*:*:stack/aws-glue*/*"`	Ermöglicht die Verwaltung von AWS CloudFormation Stacks bei der Arbeit mit Notebookservern. Namenskonvention: AWS Glue erstellt Stacks, deren Namen mit aws-glue beginnen.
`"ec2:RunInstances"`	`"arn:aws:ec2:*:*:instance/*", "arn:aws:ec2:*:*:key-pair/*", "arn:aws:ec2:*:*:image/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:subnet/*", "arn:aws:ec2:*:*:volume/*"`	Ermöglicht die Ausführung von Entwicklungsendpunkten und Notebook-Servern.
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceRole*"`	Ermöglicht AWS Glue die Übernahme der `PassRole`-Berechtigung für Rollen, die mit `AWSGlueServiceRole` beginnen.

Action (Aktion)	Ressource	Beschreibung
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceNotebookRole*"`	Ermöglicht Amazon EC2 , `PassRole` Berechtigungen für Rollen anzunehmen, die mit beginnen`AWSGlueServiceNotebookRole` .
`"iam:PassRole"`	`"arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"`	Ermöglicht AWS Glue die Übernahme der `PassRole`-Berechtigung für Rollen, die mit `service-role/AWSGlueServiceRole` beginnen.

7. Geben Sie auf dem Bildschirm „Richtlinie überprüfen“ beispielsweise einen Namen für die Richtlinie ein GlueConsoleAccessPolicy. Wenn Sie mit der Richtlinie zufrieden sind, klicken Sie auf Create policy (Richtlinie erstellen). Stellen Sie sicher, dass in dem roten Feld am oberen Bildschirmrand keine Fehler angezeigt werden. Korrigieren Sie etwaige Fehler.

> **Note**
>
> Bei ausgewählter Option Use autoformatting wird die Richtlinie neu formatiert, wenn Sie sie öffnen oder die Option Validate Policy auswählen.

Um die AWSGlue ConsoleFullAccess verwaltete Richtlinie anzuhängen

Sie können die Richtlinie `AWSGlueConsoleFullAccess` anfügen, um Berechtigungen zu erteilen, die vom AWS Glue-Konsolenbenutzer benötigt werden.

> **Note**
>
> Sie können diesen Schritt überspringen, wenn Sie Ihre eigene Richtlinie für den AWS Glue-Konsolenzugriff erstellt haben.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben dem AWSGlueConsoleFullAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).

Die von **`AWSGlueConsoleSageMakerNotebookFullAccess`** verwaltete Richtlinie anfügen

Sie können die `AWSGlueConsoleSageMakerNotebookFullAccess` Richtlinie einem Benutzer zuordnen, um SageMaker KI-Notizbücher zu verwalten, die auf der AWS Glue Konsole erstellt wurden. Zusätzlich zu anderen erforderlichen AWS Glue Konsolenberechtigungen gewährt diese Richtlinie Zugriff auf Ressourcen, die für die Verwaltung von SageMaker KI-Notizbüchern benötigt werden.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben dem AWSGlueConsoleSageMakerNotebookFullAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).

Um die CloudWatchLogsReadOnlyAccess verwaltete Richtlinie anzuhängen

Sie können die CloudWatchLogsReadOnlyAccessRichtlinie an einen Benutzer anhängen, um die von erstellten Protokolle in AWS Glue der CloudWatch Logs-Konsole anzuzeigen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben dem CloudWatchLogsReadOnlyAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).

Um die AWSCloud FormationReadOnlyAccess verwaltete Richtlinie anzuhängen

Sie können die AWSCloudFormationReadOnlyAccessRichtlinie einem Benutzer zuordnen, um die von verwendeten AWS CloudFormation Stacks AWS Glue auf der AWS CloudFormation Konsole anzuzeigen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben AWSCloudFormationReadOnlyAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).

Um die AmazonAthenaFullAccess verwaltete Richtlinie anzuhängen

Sie können die AmazonAthenaFullAccessRichtlinie an einen Benutzer anhängen, um Amazon S3 S3-Daten in der Athena-Konsole anzuzeigen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Aktivieren Sie in der Liste der Richtlinien das Kontrollkästchen neben dem AmazonAthenaFullAccess. Über das Menü Filter und das Suchfeld können Sie die Richtlinienliste filtern.
4. Klicken Sie auf Policy actions und anschließend auf Attach.
5. Wählen Sie den Benutzer aus, an den Sie die Richtlinie anfügen möchten. Über das Menü Filter und das Suchfeld können Sie die Liste der Prinzipal-Entitäten filtern. Nachdem Sie den Benutzer zum Anfügen der Richtlinie ausgewählt haben, klicken Sie auf Attach Policy (Richtlinie anfügen).

Schritt 4: Erstellen einer IAM-Richtlinie für Notebook-Server

Wenn Sie Notebooks mit Entwicklungsendpunkten verwenden möchten, müssen Sie beim Erstellen des Notebook-Servers Berechtigungen angeben. Diese Berechtigungen stellen Sie mithilfe von AWS Identity and Access Management (IAM) zur Verfügung.

Diese Richtlinie erteilt die Berechtigung für einige Amazon-S3-Aktionen zur Verwaltung von Ressourcen in Ihrem Konto, die der AWS Glue-Service benötigt, wenn er die Rolle mit dieser Richtlinie annimmt. Einige der in dieser Richtlinie angegebenen Ressourcen beziehen sich auf Standardnamen, die von AWS Glue Amazon S3-Buckets, Amazon S3-ETL-Skripts und EC2 Amazon-Ressourcen verwendet werden. Der Einfachheit halber schreibt AWS Glue standardmäßig einige Amazon-S3-Objekte in Buckets in Ihrem Konto mit dem Präfix `aws-glue-*`.

> **Note**
>
> Sie können diesen Schritt überspringen, wenn Sie die AWS verwaltete Richtlinie **`AWSGlueServiceNotebookRole`**verwenden.

In diesem Schritt erstellen Sie eine ähnliche Richtlinie wie `AWSGlueServiceNotebookRole`. Die aktuelle Version von `AWSGlueServiceNotebookRole` finden Sie in der IAM-Konsole.

So erstellen Sie eine IAM-Richtlinie für Notebooks

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Navigieren Sie auf dem Bildschirm Create Policy (Richtlinie erstellen) zu einer Registerkarte, um JSON zu bearbeiten. Erstellen Sie ein Richtliniendokument mit den folgenden JSON-Anweisungen und wählen Sie dann Review Policy (Richtlinie überprüfen) aus.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "glue:CreateDatabase",
            "glue:CreatePartition",
            "glue:CreateTable",
            "glue:DeleteDatabase",
            "glue:DeletePartition",
            "glue:DeleteTable",
            "glue:GetDatabase",
            "glue:GetDatabases",
            "glue:GetPartition",
            "glue:GetPartitions",
            "glue:GetTable",
            "glue:GetTableVersions",
            "glue:GetTables",
            "glue:UpdateDatabase",
            "glue:UpdatePartition",
            "glue:UpdateTable",
            "glue:GetJobBookmark",
            "glue:ResetJobBookmark",
            "glue:CreateConnection",
            "glue:CreateJob",
            "glue:DeleteConnection",
            "glue:DeleteJob",
            "glue:GetConnection",
```

```
            "glue:GetConnections",
            "glue:GetDevEndpoint",
            "glue:GetDevEndpoints",
            "glue:GetJob",
            "glue:GetJobs",
            "glue:UpdateJob",
            "glue:BatchDeleteConnection",
            "glue:UpdateConnection",
            "glue:GetUserDefinedFunction",
            "glue:UpdateUserDefinedFunction",
            "glue:GetUserDefinedFunctions",
            "glue:DeleteUserDefinedFunction",
            "glue:CreateUserDefinedFunction",
            "glue:BatchGetPartition",
            "glue:BatchDeletePartition",
            "glue:BatchCreatePartition",
            "glue:BatchDeleteTable",
            "glue:UpdateDevEndpoint",
            "s3:GetBucketLocation",
            "s3:ListBucket",
            "s3:ListAllMyBuckets",
            "s3:GetBucketAcl"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::crawler-public*",
            "arn:aws:s3:::aws-glue*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:DeleteObject"
         ],
         "Resource":[
```

```
            "arn:aws:s3:::aws-glue*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateTags",
            "ec2:DeleteTags"
         ],
         "Condition":{
            "ForAllValues:StringEquals":{
               "aws:TagKeys":[
                  "aws-glue-service-resource"
               ]
            }
         },
         "Resource":[
            "arn:aws:ec2:*:*:network-interface/*",
            "arn:aws:ec2:*:*:security-group/*",
            "arn:aws:ec2:*:*:instance/*"
         ]
      }
   ]
}
```

Die folgende Tabelle beschreibt die Berechtigungen, die von dieser Richtlinie erteilt werden.

Action (Aktion)	Ressource	Beschreibung
`"glue:*"`	`"*"`	Gewährt die Berechtigung zum Ausführen aller AWS Glue-API-Operationen.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl"`	`"*"`	Ermöglicht die Auflistung von Amazon-S3-Buckets über Notebook-Server.

Action (Aktion)	Ressource	Beschreibung
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Lässt das Abrufen von Amazon-S3-Objekten, die von Beispielen und Tutorials verwendet werden, aus Notebooks zu. Namenskonvention: Amazon-S3-Bucket-Namen beginnen mit crawler-public und aws-glue-.
`"s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue*"`	Ermöglicht das Speichern und Löschen von Amazon-S3-Objekte in Ihrem Konto über Notebooks. Namenskonvention: Verwendet Amazon-S3-Ordner mit dem Namen aws-glue.
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Ermöglicht das Taggen von EC2 Amazon-Ressourcen, die für Notebook-Server erstellt wurden. Namenskonvention: AWS Glue kennzeichnet EC2 Amazon-Instances mit aws-glue-service-resource.

5. Geben Sie auf Bildschirm Review Policy (Richtlinie überprüfen) Ihren Policy Name (Richtlinienname) ein, z. B. GlueServiceNotebookPolicyDefault. Geben Sie eine optionale

Beschreibung ein und wählen Sie Create policy (Richtlinie erstellen) aus, wenn Sie mit der Richtlinie zufrieden sind.

Schritt 5: Erstellen einer IAM-Rolle für Notebook-Server

Wenn Sie vorhaben, Notebooks mit Entwicklungsendpunkten zu verwenden, müssen Sie der IAM-Rolle Berechtigungen erteilen. Sie stellen diese Berechtigungen mithilfe von AWS Identity and Access Management IAM über eine IAM-Rolle bereit.

> **Note**
>
> Wenn Sie eine IAM-Rolle mithilfe der IAM-Konsole erstellen, erzeugt die Konsole automatisch ein Instance-Profil und gibt ihm denselben Namen wie der entsprechenden Rolle.

So erstellen Sie eine IAM-Rolle für Notebooks

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Wählen Sie Rolle erstellen aus.
4. Wählen Sie als Rollentyp AWS Service, suchen und wählen Sie EC2den EC2Anwendungsfall aus und wählen Sie dann Weiter: Berechtigungen aus.
5. Wählen Sie auf der Seite „Zugriffsrichtlinie anhängen“ die Richtlinien aus, die die erforderlichen Berechtigungen enthalten, z. B. AWSGlueServiceNotebookRolefür allgemeine AWS Glue Berechtigungen und die AWS verwaltete Richtlinie AmazonS3 FullAccess für den Zugriff auf Amazon S3-Ressourcen. Wählen Sie dann Next: Review aus.

   > **Note**
   >
   > Stellen Sie sicher, dass eine der Richtlinien in dieser Rolle Berechtigungen für Ihre Amazon-S3-Quellen und -Ziele erteilt. Vergewissern Sie sich außerdem, dass Ihre Richtlinie vollständigen Zugriff auf den Speicherort zulässt, an dem Sie Ihr Notebook ablegen, wenn Sie einen Notebook-Server erstellen. Sie können Ihre eigenen Richtlinien für den Zugriff auf bestimmte Amazon-S3-Ressourcen bereitstellen. Weitere Informationen zum Erstellen einer Amazon-S3-Richtlinie für Ihre Ressourcen finden Sie unter Festlegen von Ressourcen in einer Richtlinie.

Wenn Sie vorhaben, auf Amazon-S3-Quellen und -Ziele zuzugreifen, die mit SSE-KMS verschlüsselt sind, fügen Sie eine Richtlinie an, die Notebooks erlaubt, die Daten zu entschlüsseln. Weitere Informationen finden Sie unter Schutz von Daten mithilfe serverseitiger Verschlüsselung mit AWS KMS verwalteten Schlüsseln (SSE-KMS). Im Folgenden wird ein Beispiel gezeigt.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:*:111122223333:key/key-id"
            ]
        }
    ]
}
```

6. Geben Sie unter Role name (Rollenname) einen Namen für Ihre Rolle ein. Erstellen Sie die Rolle mit dem Namenspräfix `AWSGlueServiceNotebookRole`, damit die Rolle von Konsolenbenutzern an den Notebook-Server übergeben werden kann. Die von AWS Glue bereitgestellten Richtlinien erwarten, dass IAM-Servicerollen mit `AWSGlueServiceNotebookRole` beginnen. Andernfalls müssen Sie Ihren Benutzern eine Richtlinie hinzufügen, damit die `iam:PassRole`-Berechtigung für IAM-Rollen Ihren Benennungskonvention entspricht. Geben Sie z. B. ei `AWSGlueServiceNotebookRoleDefault`. Wählen Sie dann Create Role.

Schritt 6: Erstellen Sie eine IAM-Richtlinie für SageMaker KI-Notebooks

Wenn Sie SageMaker KI-Notebooks mit Entwicklungsendpunkten verwenden möchten, müssen Sie bei der Erstellung des Notebooks Berechtigungen angeben. Diese Berechtigungen stellen Sie mithilfe von AWS Identity and Access Management (IAM) zur Verfügung.

Um eine IAM-Richtlinie für SageMaker KI-Notebooks zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Richtlinien aus.
3. Wählen Sie Richtlinie erstellen aus.
4. Navigieren Sie auf der Seite Create Policy (Richtlinie erstellen) zu einer Registerkarte, um das JSON zu bearbeiten. Erstellen Sie ein Richtliniendokument mit den folgenden JSON-Anweisungen. Bearbeiten Sie *bucket-nameregion-code*, und *account-id* für Ihre Umgebung.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::bucket-name*"
            ]
        },
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents",
                "logs:CreateLogGroup"
            ],
```

```
            "Effect": "Allow",
            "Resource": [
                "arn:aws:logs:us-east-1:account-id:log-group:/aws/sagemaker/
*",
                "arn:aws:logs:us-east-1:account-id:log-group:/aws/sagemaker/
*:log-stream:aws-glue-*"
            ]
        },
        {
            "Action": [
                "glue:UpdateDevEndpoint",
                "glue:GetDevEndpoint",
                "glue:GetDevEndpoints"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:glue:us-east-1:account-id:devEndpoint/*"
            ]
        },
        {
            "Action": [
                "sagemaker:ListTags"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:sagemaker:us-east-1:account-id:notebook-instance/*"
            ]
        }
    ]
}
```

Wählen Sie dann Review policy (Richtlinie prüfen).

Die folgende Tabelle beschreibt die Berechtigungen, die von dieser Richtlinie erteilt werden.

Action (Aktion)	Ressource	Beschreibung
`"s3:ListBucket*"`	`"arn:aws:s3::: bucket-name "`	Erteilt die Berechtigung zum Auflisten von Amazon-S3-Buckets

Action (Aktion)	Ressource	Beschreibung
`"s3:GetObject"`	`"arn:aws:s3::: bucket-name *"`	Erteilt die Berechtigung zum Abrufen von Amazon S3 S3-Objekten, die von SageMaker KI-Notebooks verwendet werden.
`"logs:CreateLogStream", "logs:DescribeLogStreams", "logs:PutLogEvents", "logs:CreateLogGroup"`	`"arn:aws:logs: region-code :account-id :log-group:/aws/sagemaker/*", "arn:aws:logs: region-code :account-id :log-group:/aws/sagemaker/*:log-stream:aws-glue-*"`	Erteilt die Erlaubnis, Logs aus Notizbüchern in Amazon CloudWatch Logs zu schreiben. Namenskonvention: Schreibt Protokolle in Protokollgruppen, deren Namen mit aws-glue beginnen.
`"glue:UpdateDevEndpoint", "glue:GetDevEndpoint", "glue:GetDevEndpoints"`	`"arn:aws:glue: region-code :account-id :devEndpoint/*"`	Erteilt die Erlaubnis, einen Entwicklungsendpunkt aus SageMaker KI-Notebooks zu verwenden.
`"sagemaker:ListTags"`	`"arn:aws:sagemaker : region-code :account-id :notebook-instance /*"`	Erteilt die Erlaubnis, Tags für eine SageMaker KI-Ressource zurückzugeben. Das `aws-glue-dev-endpoint` Tag ist auf dem SageMaker KI-Notizbuch erforderlich, um das Notebook mit einem Entwicklungsendpunkt zu verbinden.

5. Geben Sie auf Bildschirm Review Policy (Richtlinie überprüfen) Ihren Policy Name (Richtlinienname) ein, z. B. `AWSGlueSageMakerNotebook`. Geben Sie eine optionale Beschreibung ein und wählen Sie Create policy (Richtlinie erstellen) aus, wenn Sie mit der Richtlinie zufrieden sind.

Schritt 7: Erstellen Sie eine IAM-Rolle für SageMaker KI-Notebooks

Wenn Sie SageMaker KI-Notebooks mit Entwicklungsendpunkten verwenden möchten, müssen Sie der IAM-Rolle Berechtigungen erteilen. Sie stellen diese Berechtigungen mithilfe von AWS Identity and Access Management (IAM) über eine IAM-Rolle bereit.

Um eine IAM-Rolle für KI-Notebooks zu erstellen SageMaker

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter. https://console.aws.amazon.com/iam/
2. Wählen Sie im linken Navigationsbereich Roles aus.
3. Wählen Sie Rolle erstellen aus.
4. Wählen Sie als Rollentyp AWS Service, suchen und wählen Sie SageMakerund wählen Sie dann den SageMaker Anwendungsfall Ausführung aus. Wählen Sie dann Next: Permissions.
5. Wählen Sie auf der Seite „Berechtigungsrichtlinie anhängen“ die Richtlinien aus, die die erforderlichen Berechtigungen enthalten, AmazonSageMakerFullAccessz. B. Wählen Sie Weiter: Prüfen aus.

   Wenn Sie vorhaben, auf Amazon-S3-Quellen und -Ziele zuzugreifen, die mit SSE-KMS verschlüsselt sind, fügen Sie eine Richtlinie an, die Notebooks erlaubt, die Daten zu entschlüsseln, wie im folgenden Beispiel gezeigt. Weitere Informationen finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung mit AWS KMS verwalteten Schlüsseln (SSE-KMS).

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:*:111122223333:key/key-id"
            ]
        }
    ]
```

```
}
```

6. Geben Sie unter Role name (Rollenname) einen Namen für Ihre Rolle ein. Verwenden Sie einen Namen, dem die Zeichenfolge vorangestellt wird, damit die Rolle von Konsolenbenutzern an SageMaker AI weitergegeben werden kann. `AWSGlueServiceSageMakerNotebookRole` AWS Gluevorausgesetzt, Richtlinien gehen davon aus, dass zunächst IAM-Rollen verwendet werden. `AWSGlueServiceSageMakerNotebookRole` Andernfalls müssen Sie Ihren Benutzern eine Richtlinie hinzufügen, damit die `iam:PassRole`-Berechtigung für IAM-Rollen Ihren Benennungskonvention entspricht.

   Geben Sie zum Beispiel `AWSGlueServiceSageMakerNotebookRole-Default` ein und klicken Sie auf Create role (Rolle erstellen).
7. Nachdem Sie die Rolle erstellt haben, fügen Sie die Richtlinie hinzu, die zusätzliche Berechtigungen ermöglicht, die für die Erstellung von SageMaker AWS Glue KI-Notebooks erforderlich sind.

   Wählen Sie die gerade erstellte Rolle `AWSGlueServiceSageMakerNotebookRole-Default` und dann Attach policies (Richtlinien anfügen). Fügen Sie die erstellte Richtlinie `AWSGlueSageMakerNotebook` der Rolle an.

AWS Beispiele für Richtlinien zur Glue-Zugriffskontrolle

Dieser Abschnitt enthält Beispiele sowohl für identitätsbasierte (IAM) Zugriffskontrollrichtlinien als auch AWS Glue Ressourcenrichtlinien.

Inhalt

- Beispiele für identitätsbasierte Richtlinien für Glue AWS
  - Bewährte Methoden für Richtlinien
  - Berechtigungen auf Ressourcenebene gelten nur für bestimmte AWS Glue-Objekte
  - Verwenden der AWS Glue-Konsole
  - Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
  - Nur-Leseberechtigung für eine Tabelle erteilen
  - Filtern Sie Tabellen nach GetTables Berechtigungen
  - Vollständiger Zugriff auf eine Tabelle und alle Partitionen gewähren
  - Steuerung des Zugriffs über Namenspräfix und explizites Verweigern
  - Zugriffssteuerung gewähren mit Tags

- Zugriff mit Tags verweigern
- Verwendung von Tags mit Listen- und Batch-API-Vorgängen
- Einstellungen über Bedingungsschlüssel oder Kontextschlüssel steuern
  - Steuern von Richtlinien, die Einstellungen über Bedingungsschlüssel steuern
  - Steuern von Richtlinien, die Einstellungen über Kontextschlüssel steuern
- Einer Identität die Möglichkeit verweigern, Datenvorschau-Sitzungen zu erstellen
- Beispiele für ressourcenbasierte Richtlinien für Glue AWS
  - Überlegungen zur Verwendung ressourcenbasierter Richtlinien mit Glue AWS
  - Verwenden einer Ressourcenrichtlinie, um den Zugriff auf das gleiche Konto zu steuern

Beispiele für identitätsbasierte Richtlinien für Glue AWS

Standardmäßig sind Benutzer und Rollen nicht berechtigt, AWS Glue-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu den von AWS Glue definierten Aktionen und Ressourcentypen, einschließlich des Formats von ARNs für jeden der Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS Glue in der Service Authorization Reference.

> **Note**
>
> Die Beispiele in diesem Abschnitt verwenden alle die `us-west-2`-Region. Sie können dies durch die AWS Region ersetzen, die Sie verwenden möchten.

Themen

- Bewährte Methoden für Richtlinien
- Berechtigungen auf Ressourcenebene gelten nur für bestimmte AWS Glue-Objekte

- Verwenden der AWS Glue-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Nur-Leseberechtigung für eine Tabelle erteilen
- Filtern Sie Tabellen nach GetTables Berechtigungen
- Vollständiger Zugriff auf eine Tabelle und alle Partitionen gewähren
- Steuerung des Zugriffs über Namenspräfix und explizites Verweigern
- Zugriffssteuerung gewähren mit Tags
- Zugriff mit Tags verweigern
- Verwendung von Tags mit Listen- und Batch-API-Vorgängen
- Einstellungen über Bedingungsschlüssel oder Kontextschlüssel steuern
- Einer Identität die Möglichkeit verweigern, Datenvorschau-Sitzungen zu erstellen

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand AWS Glue-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.

- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Berechtigungen auf Ressourcenebene gelten nur für bestimmte AWS Glue-Objekte

Eine differenziertere Kontrolle kann nur für bestimmte Objekte in AWS Glue definiert werden. Daher müssen Sie die IAM-Richtlinie Ihres Kunden so schreiben, dass API-Operationen, die Amazon Resource Names (ARNs) für die `Resource` Anweisung zulassen, nicht mit API-Vorgängen vermischt werden, die dies nicht zulassen ARNs.

Beispiel: Die folgende IAM-Richtlinie lässt API-Operationen für `GetClassifier` und `GetJobRun` zu. Sie definiert das `Resource` als*, weil Klassifikatoren und ARNs Jobausführungen AWS Glue nicht zulässig sind. Weil für bestimmte API-Operationen wie `GetDatabase` und zulässig ARNs sind`GetTable`, ARNs können sie in der zweiten Hälfte der Richtlinie angegeben werden.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetClassifier*",
                "glue:GetJobRun*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:Get*"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:123456789012:catalog",
                "arn:aws:glue:us-east-1:123456789012:database/default",
                "arn:aws:glue:us-east-1:123456789012:table/default/e*1*",
                "arn:aws:glue:us-east-1:123456789012:connection/connection2"
            ]
        }
    ]
}
```

Eine Liste der AWS Glue Objekte, die dies zulassen ARNs, finden Sie unterAWS GlueRessource angeben ARNs.

Verwenden der AWS Glue-Konsole

Um auf die AWS Glue-Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den AWS Glue-Ressourcen in Ihrem aufzulisten und anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die AWS Glue-Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die AWS Glue *ConsoleAccess* - oder *ReadOnly* AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Damit ein Benutzer mit der AWS Glue Konsole arbeiten kann, muss er über Mindestberechtigungen verfügen, die es ihm ermöglichen, mit den AWS Glue Ressourcen für sein AWS Konto zu arbeiten. Zusätzlich zu diesen AWS Glue-Berechtigungen erfordert die Konsole Berechtigungen von den folgenden Services:

- Amazon CloudWatch Logs-Berechtigungen zum Anzeigen von Protokollen.
- AWS Identity and Access Management (IAM) -Berechtigungen zum Auflisten und Weitergeben von Rollen.
- AWS CloudFormation Berechtigungen zum Arbeiten mit Stacks.
- Amazon Elastic Compute Cloud (Amazon EC2) -Berechtigungen für Listen VPCs, Subnetze, Sicherheitsgruppen, Instances und andere Objekte.
- Amazon Simple Storage Service (Amazon S3)-Berechtigungen zum Auflisten von Buckets und Objekten sowie zum Abrufen und Speichern von Skripts.
- Amazon-Redshift-Berechtigungen für die Arbeit mit Clustern.
- Amazon Relational Database Service (Amazon RDS)-Berechtigungen zum Auflisten von Instances.

Weitere Informationen zu den Berechtigungen, die Ihre Benutzer benötigen, um die AWS Glue-Konsole anzuzeigen und mit ihr zu arbeiten, finden Sie unter Schritt 3: Fügen Sie eine Richtlinie an Benutzer an, die auf AWS Glue zugreifen.

Wenn Sie eine IAM-Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Benutzer mit dieser IAM-Richtlinie. Um sicherzustellen, dass diese Benutzer die AWS Glue-Konsole weiterhin verwenden können, fügen Sie dem Benutzer auch die `AWSGlueConsoleFullAccess`-verwaltete Richtlinie, wie in AWS verwaltete (vordefinierte) Richtlinien für AWS Glue beschrieben, an.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie beinhaltet Berechtigungen, um diese Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI API oder durchzuführen. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Nur-Leseberechtigung für eine Tabelle erteilen

Die folgende Richtlinie gewährt Lesezugriff auf eine Tabelle `books` in der Datenbank `db1`. Weitere Informationen zur Ressource Amazon Resource Names (ARNs) finden Sie unterDatenkatalog ARNs.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesActionOnBooks",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables",
        "glue:GetTable"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:123456789012:catalog",
        "arn:aws:glue:us-west-2:123456789012:database/db1",
        "arn:aws:glue:us-west-2:123456789012:table/db1/books"
      ]
    }
  ]
}
```

Diese Richtlinie gewährt Lesezugriff auf eine Tabelle mit dem Namen `books` in der Datenbank namens `db1`. Zum Erteilen der Berechtigung `Get` zu einer Tabelle ist auch die Berechtigung für den Katalog und die Datenbankressourcen erforderlich.

Die folgende Richtlinie gewährt die Mindestberechtigungen zum Erstellen der Tabelle `tb1` in der Datenbank `db1`:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "glue:CreateTable"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:123456789012:table/db1/tbl1",
        "arn:aws:glue:us-west-2:123456789012:database/db1",
        "arn:aws:glue:us-west-2:123456789012:catalog"
      ]
    }
  ]
}
```

Filtern Sie Tabellen nach GetTables Berechtigungen

Angenommen, es gibt drei Tabellen `customers` , `stores` und `store_sales` in der Datenbank `db1`. Die folgende Richtlinie gewährt `GetTables` die Berechtigung für `stores` und `store_sales`, aber nicht für `customers`. Wenn Sie `GetTables` mit dieser Richtlinie aufrufen, enthält das Ergebnis nur die beiden autorisierten Tabellen (die Tabelle `customers` wird nicht zurückgegeben).

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesExample",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/store_sales",
                "arn:aws:glue:us-west-2:123456789012:table/db1/stores"
            ]
        }
    ]
}
```

Sie können die vorhergehende Richtlinie vereinfachen, indem Sie `store*` angeben und so alle Tabellennamen einschließen, die mit `store` beginnen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesExample2",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/store*"
            ]
        }
    ]
}
```

In ähnlicher Weise wird bei Verwendung von `/db1/*` zum Einschließen aller Tabellen in `db1` durch die folgende Richtlinie `GetTables` Zugriff auf alle Tabellen in `db1` gewährt.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesReturnAll",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
```

```
                "arn:aws:glue:us-west-2:123456789012:table/db1/*"
            ]
        }
    ]
}
```

Wird kein Tabellen-ARN angegeben, ist ein Aufruf von `GetTables` erfolgreich, aber es wird eine leere Liste zurückgegeben.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesEmptyResults",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1"
            ]
        }
    ]
}
```

Wenn der Datenbank-ARN in der Richtlinie fehlt, schlägt ein Aufruf von `GetTables` mit einer `AccessDeniedException` fehl.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesAccessDeny",
            "Effect": "Allow",
```

```
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:table/db1/*"
            ]
        }
    ]
}
```

Vollständiger Zugriff auf eine Tabelle und alle Partitionen gewähren

Die folgende Richtlinie gewährt alle Berechtigungen für eine Tabelle mit dem Namen books in der Datenbank db1. Dazu gehören Lese- und Schreibrechte für die Tabelle selbst, die archivierten Versionen und alle Partitionen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessOnTable",
            "Effect": "Allow",
            "Action": [
                "glue:CreateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:DeleteTableVersion",
                "glue:BatchDeleteTableVersion",
                "glue:CreatePartition",
                "glue:BatchCreatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition",
```

```
                "glue:UpdatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition"
            ],
            "Resource": [
                        "arn:aws:glue:us-west-2:123456789012:catalog",
                        "arn:aws:glue:us-west-2:123456789012:database/db1",
                        "arn:aws:glue:us-west-2:123456789012:table/db1/books"
                        ]
        }
    ]
}
```

Die vorhergehende Richtlinie kann in der Praxis vereinfacht werden.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessOnTable",
            "Effect": "Allow",
            "Action": [
                "glue:*Table*",
                "glue:*Partition*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/books"
            ]
        }
    ]
}
```

Beachten Sie, dass die minimale Granularität der differenzierten Zugriffskontrolle auf Tabellenebene liegt. Das bedeutet, dass Sie einem Benutzer keinen Zugriff auf einige Partitionen in einer Tabelle gewähren können, aber nicht auf andere, oder auf einige Tabellenspalten, aber nicht auf andere. Ein Benutzer hat entweder Zugriff auf die gesamte oder auf keinen Teil der Tabelle.

Steuerung des Zugriffs über Namenspräfix und explizites Verweigern

Nehmen wir in diesem Beispiel an, dass die Datenbanken und Tabellen in Ihrem AWS Glue-Datenkatalog anhand von Namenspräfixen organisiert sind. Die Datenbanken in der Entwicklungsphase haben das Namenspräfix `dev-` und die in Produktion haben das Namenspräfix `prod-`. Sie können die folgende Richtlinie verwenden, um Entwicklern vollen Zugriff auf alle Datenbanken, UDFs Tabellen usw. zu gewähren, die das `dev-` Präfix haben. Aber Sie gewähren Lesezugriff auf alle Elemente mit dem Präfix `prod-`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DevAndProdFullAccess",
            "Effect": "Allow",
            "Action": [
                "glue:*Database*",
                "glue:*Table*",
                "glue:*Partition*",
                "glue:*UserDefinedFunction*",
                "glue:*Connection*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/dev-*",
                "arn:aws:glue:us-west-2:123456789012:database/prod-*",
                "arn:aws:glue:us-west-2:123456789012:table/dev-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/dev-*",
                "arn:aws:glue:us-west-2:123456789012:table/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/dev-*/
*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/dev-
*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/prod-*/
*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/prod-
*",
                "arn:aws:glue:us-west-2:123456789012:connection/dev-*",
                "arn:aws:glue:us-west-2:123456789012:connection/prod-*"
```

```
            ]
        },
        {
            "Sid": "ProdWriteDeny",
            "Effect": "Deny",
            "Action": [
                "glue:*Create*",
                "glue:*Update*",
                "glue:*Delete*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:database/prod-*",
                "arn:aws:glue:us-west-2:123456789012:table/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/prod-*/
*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/prod-
*",
                "arn:aws:glue:us-west-2:123456789012:connection/prod-*"
            ]
        }
    ]
}
```

Die zweite Anweisung in der vorhergehenden Richtlinie verwendet explizit `deny`. Sie können explizit `deny` verwenden, um beliebige `allow`-Berechtigungen zu überschreiben, die dem Prinzipal erteilt werden. Auf diese Weise können Sie den Zugriff auf kritische Ressourcen sperren und verhindern, dass eine andere Richtlinie versehentlich Zugriff auf diese gewährt.

Auch wenn die erste Anweisung im vorherigen Beispiel Vollzugriff auf `prod`--Ressourcen gewährt, widerruft die zweite Anweisung explizit den Schreibzugriff auf sie, sodass nur Lesezugriff auf `prod`--Ressourcen besteht.

Zugriffssteuerung gewähren mit Tags

Angenommen, Sie möchten den Zugriff auf einen `t2`-Auslöser auf einen bestimmten Benutzer mit dem Namen `Tom` in Ihrem Konto einschränken. Alle anderen Benutzer, einschließlich `Sam`, haben Zugriff auf den Auslöser `t1`. Die Auslöser `t1` und `t2` haben die folgenden Eigenschaften.

```
aws glue get-triggers
{
```

```
    "Triggers": [
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t1",
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        },
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t2",
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        }
    ]
}
```

Der AWS Glue-Administrator hat den Tag-Wert `Tom` (`aws:ResourceTag/Name": "Tom"`) an den Auslöser `t2` angehängt. Außerdem hat der AWS Glue-Administrator Tom eine IAM-Richtlinie mit einer auf dem Tag basierenden Bedingungsanweisung zugewiesen. Dies hat zur Folge, dass Tom nur AWS Glue-Operationen verwenden kann, die auf Ressourcen mit dem Tag-Wert `Tom` einwirken.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
         "Effect": "Allow",
         "Action": "glue:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
```

```
                "aws:ResourceTag/Name": "Tom"
            }
         }
        }
    ]
}
```

Wenn Tom versucht, auf den Auslöser `t1` zuzugreifen, erhält er die Meldung, dass ihm der Zugriff verweigert wurde. Er kann aber den Auslöser `t2` erfolgreich abrufen.

```
aws glue get-trigger --name t1

An error occurred (AccessDeniedException) when calling the GetTrigger operation:
 User: Tom is not authorized to perform: glue:GetTrigger on resource: arn:aws:glue:us-
east-1:123456789012:trigger/t1

aws glue get-trigger --name t2
{
    "Trigger": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j1"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Tom kann den pluralen `GetTriggers`-API-Vorgang zum Auflisten von Auslösern nicht verwenden, da dieser Vorgang keine Filterung nach Tags unterstützt.

Um Tom Zugriff auf `GetTriggers` zu gewähren, erstellt der AWS Glue-Administrator eine Richtlinie, die die Berechtigungen in zwei Abschnitte unterteilt. Ein Abschnitt gewährt Tom Zugriff auf alle Auslöser mit der `GetTriggers`-API-Operation. Der zweite Abschnitt gewährt Tom Zugriff auf API-Operationen, die mit dem Wert `Tom` markiert sind. Mit dieser Richtlinie wird Tom sowohl Zugriff auf `GetTriggers` als auch auf `GetTrigger` gewährt, um auf den Auslöser `t2` zuzugreifen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "glue:GetTriggers",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "glue:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Name": "Tom"
                }
            }
        }
    ]
}
```

Zugriff mit Tags verweigern

Ein anderer Ansatz einer Ressourcenrichtlinie besteht darin, den Zugriff auf Ressourcen ausdrücklich zu verweigern.

> **Important**
>
> Eine explizite Verweigerungsrichtlinie funktioniert nicht für plurale APIs wie `GetTriggers`.

In der folgenden Beispielrichtlinie sind alle AWS Glue-Auftragsvorgänge zulässig. Die zweite `Effect`-Aussage verweigert jedoch ausdrücklich den Zugriff auf Aufträge, die mit dem `Team`-Schlüssel und dem `Special`-Wert gekennzeichnet sind.

Wenn ein Administrator einer Identität die folgende Richtlinie zuordnet, kann die Identität auf alle Aufträge zugreifen, mit Ausnahme derjenigen, die mit dem `Team`-Schlüssel und dem `Special`-Wert gekennzeichnet sind.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "glue:*",
      "Resource": "arn:aws:glue:us-east-1:123456789012:job/*"
    },
    {
      "Effect": "Deny",
      "Action": "glue:*",
      "Resource": "arn:aws:glue:us-east-1:123456789012:job/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Team": "Special"
        }
      }
    }
  ]
}
```

Verwendung von Tags mit Listen- und Batch-API-Vorgängen

Ein dritter Ansatz für das Schreiben einer Ressourcenrichtlinie besteht darin, den Zugriff auf Ressourcen mithilfe einer `List`-API-Operation zum Auflisten von Ressourcen für einen Tag-Wert zuzulassen. Anschließend verwenden Sie die entsprechende `Batch`-API-Operation, um Zugriff auf Details zu spezifischen Ressourcen zu gewähren. Bei diesem Ansatz muss der Administrator nicht den Zugriff auf die pluralen API-Operationen `GetCrawlers`, `GetDevEndpoints`, `GetJobs` oder `GetTriggers` zulassen. Stattdessen können Sie das Auflisten der Ressourcen mit den folgenden API-Operationen gewähren:

- `ListCrawlers`
- `ListDevEndpoints`
- `ListJobs`
- `ListTriggers`

Und Sie können das Anfordern von Details zu einzelnen Ressourcen mit den folgenden API-Operationen gewähren:

- `BatchGetCrawlers`
- `BatchGetDevEndpoints`
- `BatchGetJobs`
- `BatchGetTriggers`

Um diesen Ansatz zu nutzen, können Sie als Administrator wie folgt vorgehen:

1. Hinzufügen von Tags zu Ihren Crawlern, Entwicklungsendpunkten, Aufträgen und Auslösern.
2. Verweigern des Benutzerzugriffs auf `Get`-API-Operationen wie `GetCrawlers`, `GetDevEndponts`, `GetJobs` und `GetTriggers`.
3. Damit Benutzer herauszufinden können, auf welche markierten Ressourcen sie zugreifen können, lassen Sie den Benutzerzugriff auf `List`-API-Operationen wie `ListCrawlers`, `ListDevEndponts`, `ListJobs` und `ListTriggers` zu.
4. Verweigern Sie Benutzern den Zugriff auf AWS Glue Tagging APIs, z. B. `TagResource` und`UntagResource`.
5. Gewähren des Benutzerzugriffs auf Ressourcendetails mit `BatchGet`-API-Operationen wie `BatchGetCrawlers`, `BatchGetDevEndponts`, `BatchGetJobs` und `BatchGetTriggers`.

Wenn Sie beispielsweise die `ListCrawlers`-Operation aufrufen, geben Sie einen Tag-Wert ein, der dem Benutzernamen entspricht. Das Ergebnis ist dann eine Liste der Crawler, die mti den angegebenen Tag-Werten übereinstimmen. Stellen Sie `BatchGetCrawlers` die Liste mit den Namen der Crawler zur Verfügung, um detaillierte Informationen zu jedem Crawler mit dem angegebenen Tag zu erhalten.

Wenn es Tom beispielsweise möglich sein soll, Details zu Auslösern abzurufen, die mit `Tom` markiert sind, kann der Administrator Tags zu Auslösern für `Tom` hinzufügen, allen Benutzern den Zugriff auf die `GetTriggers`-API-Operation verweigern und allen Benutzern den Zugriff auf `ListTriggers` und `BatchGetTriggers` gewähren.

Hier ist die Ressourcen-Richtlinie, die der AWS Glue-Administrator Tom gewährt. Im ersten Teil der Richtlinie werden AWS Glue-API-Operationen für `GetTriggers` verweigert. Im zweiten Teil der Richtlinie wird `ListTriggers` für alle Ressourcen gewährt. Im dritten Teil wird den durch `Tom` markierten Ressourcen jedoch Zugriff mit `BatchGetTriggers`-Zugriff gewährt.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "glue:GetTriggers",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListTriggers"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:BatchGetTriggers"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Name": "Tom"
                }
            }
        }
    ]
}
```

Unter Verwendung der gleichen Auslöser wie im vorherigen Beispiel kann Tom auf den Auslöser `t2`, aber nicht auf den Auslöser `t1` zugreifen. Das folgende Beispiel zeigt die Ergebnisse, wenn Tom versucht, mit `BatchGetTriggers` auf `t1` und `t2` zuzugreifen.

```
aws glue batch-get-triggers --trigger-names t2
```

```
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j2"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}

aws glue batch-get-triggers  --trigger-names t1

An error occurred (AccessDeniedException) when calling the BatchGetTriggers operation:
 No access to any requested resource.
```

Das folgende Beispiel zeigt die Ergebnisse, wenn Tom in demselben `BatchGetTriggers`-Aufruf versucht, sowohl auf den Auslöser `t2` als auch auf den Auslöser `t3` (nicht vorhanden) zuzugreifen. Beachten Sie, dass nur `t2` zurückgegeben wird, da Tom Zugriff auf Auslöser `t2` hat und dieser vorhanden ist. Obwohl Tom auf Auslöser `t3` zugreifen darf, wird `t3` in der Antwort in einer Liste von `"TriggersNotFound": []` zurückgegeben, da Auslöser `t3` nicht vorhanden ist.

```
aws glue batch-get-triggers --trigger-names t2 t3
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j2"
            }
        ],
        "TriggersNotFound": ["t3"],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Einstellungen über Bedingungsschlüssel oder Kontextschlüssel steuern

Sie können Bedingungsschlüssel oder Kontextschlüssel verwenden, wenn Sie Berechtigungen zum Erstellen und Aktualisieren von Aufträgen erteilen. In diesen Abschnitten werden die Schlüssel behandelt:

- Steuern von Richtlinien, die Einstellungen über Bedingungsschlüssel steuern
- Steuern von Richtlinien, die Einstellungen über Kontextschlüssel steuern

Steuern von Richtlinien, die Einstellungen über Bedingungsschlüssel steuern

AWS Glue bietet drei IAM-Zustandsschlüssel `glue:VpcIdsglue:SubnetIds`, und`glue:SecurityGroupIds`. Sie können die Bedingungsschlüssel in IAM-Richtlinien verwenden, wenn Sie Berechtigungen zum Erstellen und Aktualisieren von Aufträgen erteilen. Mit dieser Einstellung können Sie sicherstellen, dass Aufträge oder Sitzungen nicht für die Ausführung außerhalb einer gewünschten VPC-Umgebung erstellt (oder aktualisiert) werden. Die Informationen zu den VPC-Einstellungen stammen nicht direkt aus der `CreateJob`-Anfrage, sondern werden aus dem Auftragsfeld „connections“ abgeleitet, das auf eine AWS Glue-Verbindung verweist.

Beispielverwendung

Erstellen Sie eine AWS Glue Netzwerkverbindung mit dem Namen "traffic-monitored-connection" mit den gewünschten Werten VpcId „vpc-id1234", und. SubnetIds SecurityGroupIds

Geben Sie die Bedingungsschlüssel-Bedingung für die `CreateJob`-und `UpdateJob`-Aktion in der IAM-Richtlinie.

```
{
  "Effect": "Allow",
  "Action": [
    "glue:CreateJob",
    "glue:UpdateJob"
  ],
  "Resource": [
    "*"
  ],
  "Condition": {
    "ForAnyValue:StringLike": {
      "glue:VpcIds": [
        "vpc-id1234"
```

```
      ]
    }
  }
}
```

Sie können eine ähnliche IAM-Richtlinie erstellen, um das Erstellen eines AWS Glue-Auftrags ohne Angabe von Verbindungsinformationen zu verhindern.

Beschränken von Sitzungen auf VPCs

Um die Ausführung erstellter Sitzungen innerhalb einer bestimmten VPC zu erzwingen, schränken Sie die Rollenberechtigung ein, indem Sie einen `Deny`-Effekt auf die `glue:CreateSession`-Aktion hinzufügen, mit der Bedingung, dass „glue:vpc-id“ nicht gleich „vpc-<123>“ ist. Zum Beispiel:

```
"Effect": "Deny",
"Action": [
    "glue:CreateSession"
 ],
"Condition": {
    "StringNotEquals" : {"glue:VpcIds" : ["vpc-123"]}
}
```

Sie können auch erzwingen, dass erstellte Sitzungen innerhalb einer VPC ausgeführt werden, indem Sie einen `Deny`-Effekt auf die `glue:CreateSession`-Aktion mit der Bedingung hinzufügen, dass das `glue:vpc-id` Null ist. Zum Beispiel:

```
{
    "Effect": "Deny",
    "Action": [
        "glue:CreateSession"
    ],
      "Condition": {
        "Null": {"glue:VpcIds": true}
    }
},
{
    "Effect": "Allow",
    "Action": [
        "glue:CreateSession"
    ],
```

```
    "Resource": ["*"]
}
```

Steuern von Richtlinien, die Einstellungen über Kontextschlüssel steuern

AWS Glue stellt für jede Rollensitzung einen Kontextschlüssel (`glue:CredentialIssuingService= glue.amazonaws.com`) bereit, AWS Glue der für den Job und den Entwickler-Endpunkt verfügbar ist. Auf diese Weise können Sie Sicherheitskontrollen für die von AWS Glue Skripten ausgeführten Aktionen implementieren. AWS Gluestellt für jede Rollensitzung, in AWS Glue der im Namen des Kunden (nicht über einen job/dev Endpunkt, sondern direkt durch den AWS Service`glue:RoleAssumedBy=glue.amazonaws.com`) ein anderer Service angerufen wird, einen weiteren Kontextschlüssel () bereit. AWS Glue

Beispielverwendung

Geben Sie die bedingte Berechtigung in einer IAM-Richtlinie an und fügen Sie diese an die Rolle an, die von einem AWS Glue-Auftrag verwendet wird. Dadurch wird sichergestellt, dass bestimmte Aktionen davon allowed/denied abhängen, ob die Rollensitzung für eine AWS Glue Job-Laufzeitumgebung verwendet wird.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

Einer Identität die Möglichkeit verweigern, Datenvorschau-Sitzungen zu erstellen

Dieser Abschnitt enthält ein Beispiel für eine IAM-Richtlinie, mit der einer Identität die Möglichkeit verweigert wird, Datenvorschau-Sitzungen zu erstellen. Verknüpfen Sie diese Richtlinie mit der Identität, die unabhängig von der Rolle ist, die die Datenvorschau-Sitzung während ihrer Ausführung verwendet.

```
{
```

```
    "Sid": "DatapreviewDeny",
    "Effect": "Deny",
     "Action": [
           "glue:CreateSession"
     ],
     "Resource": [
          "arn:aws:glue:*:*:session/glue-studio-datapreview*"
      ]
}
```

Beispiele für ressourcenbasierte Richtlinien für Glue AWS

Dieser Abschnitt enthält beispielhafte Ressourcenrichtlinien, einschließlich Richtlinien, die kontoübergreifenden Zugriff gewähren.

In den Beispielen wird das AWS Command Line Interface (AWS CLI) verwendet, um mit zu interagieren AWS Glue Service-API-Operationen. Sie können dieselben Operationen auf dem ausführen AWS Glue Konsole oder mit einem der AWS SDKs.

> **Important**
>
> Durch Ändern eines AWS Glue Ressourcenrichtlinie, Sie könnten versehentlich Berechtigungen für bestehende Benutzer entziehen AWS Glue Benutzer in Ihrem Konto und verursachen unerwartete Störungen. Probieren Sie diese Beispiele nur in Entwicklungs- oder Testkonten aus und stellen Sie sicher, dass sie keine bestehenden Arbeitsabläufe unterbrechen, bevor Sie die Änderungen vornehmen.

Themen

- Überlegungen zur Verwendung ressourcenbasierter Richtlinien mit Glue AWS
- Verwenden einer Ressourcenrichtlinie, um den Zugriff auf das gleiche Konto zu steuern

Überlegungen zur Verwendung ressourcenbasierter Richtlinien mit Glue AWS

> **Note**
>
> Sowohl IAM-Richtlinien als auch ein AWS Glue Die Verbreitung der Ressourcenrichtlinie dauert einige Sekunden. Nachdem Sie eine neue Richtlinie angefügt haben, stellen Sie

möglicherweise fest, dass die alte Richtlinie noch in Kraft ist, bis die neue Richtlinie durch das System verbreitet wurde.

Sie verwenden ein im JSON-Format geschriebenes Richtliniendokument, um eine Ressourcenrichtlinie zu erstellen oder zu ändern. Die Richtliniensyntax ist die gleiche wie bei einer identitätsbasierten IAM-Richtlinie (siehe IAM-JSON-Richtlinienreferenz), mit den folgenden Ausnahmen:

- Ein `"Principal"`- oder `"NotPrincipal"`-Block ist für jede Richtlinienanweisung erforderlich.
- Das `"Principal"` oder `"NotPrincipal"` muss gültige bestehende Prinzipale identifizieren. Platzhalter-Muster (z. B. `arn:aws:iam::`*`account-id`*`:user/*`) sind nicht erlaubt.
- Der `"Resource"` Block in der Richtlinie erfordert, dass alle Ressourcen ARNs der folgenden Syntax für reguläre Ausdrücke entsprechen (wobei die *`region`* erste `%s` die und die zweite die `%s` ist*`account-id`*):

  ```
  *arn:aws:glue:%s:%s:(\*|[a-zA-Z\*]+\/?.*)
  ```

  Zum Beispiel sind sowohl `arn:aws:glue:us-west-2:`*`account-id`*`:*` als auch `arn:aws:glue:us-west-2:`*`account-id`*`:database/default` erlaubt, aber `*` ist nicht erlaubt.
- Im Gegensatz zu identitätsbasierten Richtlinien ist ein AWS Glue Die Ressourcenrichtlinie darf nur Amazon-Ressourcennamen (ARNs) von Ressourcen enthalten, die zu dem Katalog gehören, an den die Richtlinie angehängt ist. Solche beginnen ARNs immer mit`arn:aws:glue:`.
- Eine Richtlinie kann nicht dazu führen, dass die Identität, die sie erstellt, von der weiteren Erstellung oder Änderung von Richtlinien ausgeschlossen wird.
- Ein JSON-Dokument für Ressourcenrichtlinien darf eine Größe von 10 kB nicht überschreiten.

Verwenden einer Ressourcenrichtlinie, um den Zugriff auf das gleiche Konto zu steuern

In diesem Beispiel erstellt ein Admin-Benutzer in Konto A eine Ressourcenrichtlinie, die dem IAM-Benutzer `Alice` in Konto A Vollzugriff auf den Katalog gewährt. Alice hat keine angefügte IAM-Richtlinie.

Dazu führt der Admin-Benutzer den folgenden AWS CLI Befehl aus.

```
# Run as admin of Account A
```

```
$ aws glue put-resource-policy --profile administrator-name --region us-west-2 --
policy-in-json '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
  ]
}'
```

Anstatt das JSON-Richtliniendokument als Teil Ihres AWS CLI Befehls einzugeben, können Sie ein Richtliniendokument in einer Datei speichern und im AWS CLI Befehl auf den Dateipfad mit dem Präfix verweisen. `file://` Im Folgenden finden Sie ein Beispiel dafür, wie Sie dies tun können.

```
$ echo '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
  ]
```

```
}' > /temp/policy.json

$ aws glue put-resource-policy --profile admin1 \
    --region us-west-2 --policy-in-json file:///temp/policy.json
```

Nachdem diese Ressourcenrichtlinie verbreitet wurde, kann Alice auf alle zugreifen AWS Glue Ressourcen in Konto A, wie folgt.

```
# Run as user Alice
$ aws glue create-database --profile alice --region us-west-2 --database-input '{
    "Name": "new_database",
    "Description": "A new database created by Alice",
    "LocationUri": "s3://amzn-s3-demo-bucket"
}'

$ aws glue get-table --profile alice --region us-west-2 --database-name "default" --
table-name "tbl1"}
```

Als Antwort auf Alices `get-table` Anruf AWS Glue Der Dienst gibt Folgendes zurück.

```
{
  "Table": {
    "Name": "tbl1",
    "PartitionKeys": [],
    "StorageDescriptor": {
        ......
    },
    ......
  }
}
```

Gewährung AWS verwalteter Richtlinien für AWS Glue

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet wird AWS. AWS Verwaltete Richtlinien sind so konzipiert, dass sie Berechtigungen für viele gängige Anwendungsfälle bereitstellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie für alle AWS Kunden verfügbar

sind. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

AWS verwaltete (vordefinierte) Richtlinien für AWS Glue

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet AWS werden. Diese AWS verwalteten Richtlinien gewähren die erforderlichen Berechtigungen für allgemeine Anwendungsfälle, sodass Sie nicht erst untersuchen müssen, welche Berechtigungen benötigt werden. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien im IAM-Benutzerhandbuch.

Die folgenden AWS verwalteten Richtlinien, die Sie Identitäten in Ihrem Konto zuordnen können, sind spezifisch für ein Anwendungsszenario AWS Glue und nach Anwendungsszenarien gruppiert:

- AWSGlueConsoleFullAccess— Gewährt vollen Zugriff auf AWS Glue Ressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, die AWS Management Console verwendet. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird typischerweise mit Benutzern der AWS Glue-Konsole verknüpft.
- AWSGlueServiceRole— Gewährt Zugriff auf Ressourcen, die für die Ausführung verschiedener AWS Glue Prozesse in Ihrem Namen erforderlich sind. Zu diesen Ressourcen gehören AWS Glue Amazon S3, IAM, CloudWatch Logs und Amazon EC2. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben AWS Glue-Prozesse die erforderlichen Berechtigungen. Diese Richtlinie wird typischerweise mit Rollen verknüpft, die bei der Definition von Crawlern, Aufträgen und Entwicklungsendpunkten angegeben werden.
- AwsGlueSessionUserRestrictedServiceRole— Bietet vollen Zugriff auf alle AWS Glue Ressourcen mit Ausnahme von Sitzungen. Sie erlaubt Benutzern nur die interaktiven Sitzungen zu erstellen und zu verwenden, die mit dem Benutzer verknüpft sind. Diese Richtlinie umfasst weitere Berechtigungen, die für AWS Glue die Verwaltung von AWS Glue Ressourcen in anderen AWS Diensten erforderlich sind. Die Richtlinie ermöglicht auch das Hinzufügen von Tags zu AWS Glue Ressourcen in anderen AWS Diensten.

> **Note**
>
> Um die vollen Sicherheitsvorteile zu erzielen, gewähren Sie diese Richtlinie nicht einem Benutzer, dem die `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, oder `AWSGlueConsoleSageMakerNotebookFullAccess`-Richtlinie zugewiesen wurde.

- AwsGlueSessionUserRestrictedPolicy— Ermöglicht den Zugriff auf die Erstellung AWS Glue interaktiver Sitzungen mithilfe der `CreateSession` API-Operation nur, wenn ein Tag-Schlüssel „Eigentümer“ und ein Wert angegeben werden, die mit der AWS Benutzer-ID des Beauftragten übereinstimmen. Diese Identitätsrichtlinie ist dem IAM-Benutzer zugeordnet, der der `CreateSession`-API-Vorgang aufruft. Diese Richtlinie ermöglicht es dem Beauftragten auch, mit den AWS Glue interaktiven Sitzungsressourcen zu interagieren, die mit einem „Eigentümer“ -Tag und einem Wert erstellt wurden, die seiner Benutzer-ID entsprechen. AWS Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „Eigentümer“-Tags einer AWS GlueSitzungsressource nach dem Erstellen der Sitzung.

  > **Note**
  >
  > Um die vollen Sicherheitsvorteile zu erzielen, gewähren Sie diese Richtlinie nicht einem Benutzer, dem die `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, oder `AWSGlueConsoleSageMakerNotebookFullAccess`-Richtlinie zugewiesen wurde.

- AwsGlueSessionUserRestrictedNotebookServiceRole— Bietet ausreichend Zugriff auf die AWS Glue Studio Notebook-Sitzung, um mit bestimmten AWS Glue interaktiven Sitzungsressourcen zu interagieren. Dabei handelt es sich um Ressourcen, die mit dem Tagwert „owner“ erstellt wurden, der der AWS Benutzer-ID des Prinzipals (IAM-Benutzer oder Rolle) entspricht, der das Notizbuch erstellt. Weitere Informationen zu diesen Tags finden Sie im Diagramm Prinzipal-Schlüsselwerte im IAM-Benutzerhandbuch.

  Diese Servicerollenrichtlinie ist an die Rolle angehängt, die mit einer Magic-Anweisung innerhalb des Notebooks angegeben oder als Rolle an den `CreateSession`-API-Vorgang gegeben wird. Diese Richtlinie erlaubt es dem Prinzipal außerdem, nur dann eine AWS Glue interaktive Sitzung von der AWS Glue Studio Notebook-Oberfläche aus zu erstellen, wenn der Tag-Schlüssel „Besitzer“ und der Wert mit der AWS Benutzer-ID des Prinzipals übereinstimmen. Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „Eigentümer“-Tags einer AWS GlueSitzungsressource nach dem Erstellen der Sitzung. Diese Richtlinie umfasst auch Berechtigungen zum Schreiben und Lesen aus Amazon S3 S3-Buckets, zum Schreiben von

CloudWatch Protokollen und zum Erstellen und Löschen von Tags für EC2 Amazon-Ressourcen, die von AWS Glue verwendet werden.

> **Note**
>
> Um die vollen Sicherheitsvorteile zu erzielen, gewähren Sie diese Richtlinie nicht einer Rolle, welcher die `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, oder `AWSGlueConsoleSageMakerNotebookFullAccess`-Richtlinie zugewiesen wurde.

- AwsGlueSessionUserRestrictedNotebookPolicy— Ermöglicht den Zugriff auf die Erstellung einer AWS Glue interaktiven Sitzung über die AWS Glue Studio Notebook-Oberfläche nur, wenn es einen Tag-Schlüssel „Besitzer“ und einen Wert gibt, die dem AWS Benutzer und IDof dem Prinzipal (IAM-Benutzer oder Rolle) entsprechen, der das Notizbuch erstellt. Weitere Informationen zu diesen Tags finden Sie im Diagramm Prinzipal-Schlüsselwerte im IAM-Benutzerhandbuch.

  Diese Richtlinie ist an den Prinzipal (IAM-Benutzer oder -Rolle) angehängt, der Sitzungen von der AWS Glue Studio-Notebook-Schnittstelle erstellt. Diese Richtlinie ermöglicht auch ausreichenden Zugriff auf das AWS Glue Studio-Notebook, um mit bestimmten interaktiven Sitzungsressourcen von AWS Glue zu interagieren. Dabei handelt es sich um Ressourcen, die mit dem Tagwert „owner“ erstellt wurden, der der AWS Benutzer-ID des Prinzipals entspricht. Diese Richtlinie verweigert die Berechtigung zum Ändern oder Entfernen von „Eigentümer“-Tags einer AWS GlueSitzungsressource nach dem Erstellen der Sitzung.
- AWSGlueServiceNotebookRole— Gewährt Zugriff auf AWS Glue Sitzungen, die in einem AWS Glue Studio Notizbuch gestartet wurden. Diese Richtlinie ermöglicht das Auflisten und Abrufen von Sitzungsinformationen für alle Sitzungen, erlaubt Benutzern jedoch nur, Sitzungen zu erstellen und zu verwenden, die mit ihrer AWS Benutzer-ID gekennzeichnet sind. Diese Richtlinie verweigert die Erlaubnis, „Besitzer“ -Tags aus AWS Glue Sitzungsressourcen zu ändern oder zu entfernen, die mit ihrer AWS ID gekennzeichnet sind.

  Weisen Sie diese Richtlinie dem AWS Benutzer zu, der Jobs über die Notebook-Oberfläche in AWS Glue Studio erstellt.
- AWSGlueConsoleSageMakerNotebookFullAccess— Gewährt vollen Zugriff auf AWS Glue- und SageMaker KI-Ressourcen, wenn die Identität, an die die Richtlinie angehängt ist, die verwendet AWS Management Console. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole zugewiesen, die SageMaker KI-Notizbücher verwalten.

- AWSGlueSchemaRegistryFullAccess— Gewährt vollen Zugriff auf AWS Glue Schemaregistrierungsressourcen, wenn die Identität, an die die Richtlinie angehängt ist, das AWS Management Console Oder verwendet AWS CLI. Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole oder Benutzern zugewiesen AWS CLI , die die AWS Glue Schemaregistrierung verwalten.
- AWSGlueSchemaRegistryReadonlyAccess— Gewährt schreibgeschützten Zugriff auf AWS Glue Schemaregistrierungsressourcen, wenn eine Identität, an die die Richtlinie angehängt ist, das AWS Management Console Oder verwendet. AWS CLI Wenn Sie die Namenskonvention für Ressourcen befolgen, die in dieser Richtlinie angegeben sind, haben Benutzer alle Konsolenfunktionalitäten. Diese Richtlinie wird in der Regel Benutzern der AWS Glue Konsole oder Benutzern der AWS CLI AWS Glue Schemaregistry zugewiesen.

> **Note**
>
> Sie können diese Berechtigungsrichtlinien prüfen, indem Sie sich bei der IAM-Konsole anmelden und dort nach bestimmten Richtlinien suchen.

Sie können auch Ihre eigenen, benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für AWS -Glue-Aktionen und -Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen.

Um eine Verbindung mit VPC-Konfiguration herzustellen und gleichzeitig eine benutzerdefinierte IAM-Rolle zu verwenden, muss sie über die folgenden VPC-Zugriffsaktionen verfügen:

- Secretsmanager: GetSecretValue
- Geheimnismanager: PutSecretValue
- Geheimnismanager: DescribeSecret
- ec2: CreateNetworkInterface
- ec2: DeleteNetworkInterface
- ec2: DescribeNetworkInterfaces
- ec2: DescribeSubnets

AWS Glue in AWS verwaltete Richtlinien einbinden

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für AWS Glue an, seit dieser Dienst begonnen hat, diese Änderungen zu verfolgen. Um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Seite mit dem Verlauf der AWS Glue-Dokumente.

Änderung	Beschreibung	Datum
AwsGlueSessionUserRestrictedNotebookPolicy — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie „`glue:TagResource` Aktion zulassen“ für den Besitzer-Tag-Schlüssel hinzu. Erforderlich für die Unterstützung von tag-on-create On-For-Sitzungen mit Eigentümer-Tag-Schlüssel.	30. August 2024
AwsGlueSessionUserRestrictedNotebookServiceRole — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie „`glue:TagResource` Aktion zulassen“ für den Besitzer-Tag-Schlüssel hinzu. Erforderlich für die Unterstützung von tag-on-create On-For-Sitzungen mit Eigentümer-Tag-Schlüssel.	30. August 2024
AwsGlueSessionUserRestrictedPolicy — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie „`glue:TagResource` Aktion zulassen“ für den Besitzer-Tag-Schlüssel hinzu. Erforderlich für die Unterstützung von tag-on-create On-For-Sitzungen mit Eigentümer-Tag-Schlüssel.	5. August 2024
AwsGlueSessionUserRestrictedServiceRole — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie „`glue:TagResource` Aktion zulassen“ für den Besitzer-Tag-Schlüssel hinzu. Erforderlich für die Unterstützung von tag-on-cr	5. August 2024

Änderung	Beschreibung	Datum
	eate On-For-Sitzungen mit Eigentümer-Tag-Schlüssel.	
AwsGlueSessionUserRestricted Policy — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `glue:StartCompletion` und `glue:GetCompletion` zur Richtlinie hinzu. Erforderlich für die Amazon Q-Datenintegration in AWS Glue.	30. April 2024
AwsGlueSessionUserRestricted NotebookServiceRole — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `glue:StartCompletion` und `glue:GetCompletion` zur Richtlinie hinzu. Erforderlich für die Amazon Q-Datenintegration in AWS Glue.	30. April 2024
AwsGlueSessionUserRestricted ServiceRole — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `glue:StartCompletion` und `glue:GetCompletion` zur Richtlinie hinzu. Erforderlich für die Amazon Q-Datenintegration in AWS Glue.	30. April 2024
AWSGlueServiceNotebookRole — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `glue:StartCompletion` und `glue:GetCompletion` zur Richtlinie hinzu. Erforderlich für die Amazon Q-Datenintegration in AWS Glue.	30. Januar 2024

Änderung	Beschreibung	Datum
AwsGlueSessionUser RestrictedNotebookPolicy — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `glue:StartCompletion` und `glue:GetCompletion` zur Richtlinie hinzu. Erforderlich für die Amazon Q-Datenintegration in AWS Glue.	29. November 2023
AWSGlueServiceNotebookRole — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `codewhisperer:GenerateRecommendations` zur Richtlinie hinzu. Erforderlich für eine neue Funktion, bei der AWS Glue CodeWhisperer Empfehlungen generiert.	9. Oktober 2023
AWSGlueServiceRole — Kleinere Aktualisierung einer bestehenden Richtlinie.	Beschränken Sie den Umfang der CloudWatch Berechtigungen, um die AWS Glue-Protokollierung besser widerzuspiegeln.	04. August 2023
AWSGlueConsoleFullAccess — Kleinere Aktualisierung einer bestehenden Richtlinie.	Fügen Sie der Richtlinie `databrew`-Rezeptlisten- und Beschreibungsberechtigungen hinzu. Erforderlich, um vollen Administratorzugriff für neue Funktionen bereitzustellen, mit denen AWS Glue auf Rezepte zugreifen kann.	09. Mai 2023

Änderung	Beschreibung	Datum
AWSGlueConsoleFullAccess — Geringfügige Aktualisierung einer bestehenden Richtlinie.	Fügen Sie `cloudformation:ListStacks` zur Richtlinie hinzu. Behält die vorhandenen Funktionen nach Änderungen der AWS CloudFormation Autorisierungsanforderungen bei.	28. März 2023
Neue verwaltete Richtlinien für das Interactive-Sessions-Feature hinzugefügt • AwsGlueSessionUserRestrictedServiceRole • AwsGlueSessionUserRestrictedPolicy • AwsGlueSessionUserRestrictedNotebookServiceRole • AwsGlueSessionUserRestrictedNotebookPolicy	Diese Richtlinien wurden entwickelt, um zusätzliche Sicherheit für Interactive Sessions und Notebooks in AWS Glue Studio zu bieten. Die Richtlinien beschränken den Zugriff auf den `CreateSession` -API-Vorgang, damit nur der Eigentümer Zugriff hat.	30. November 2021

Änderung	Beschreibung	Datum
AWSGlueConsoleSageMakerNotebookFullAccess — Aktualisierung einer bestehenden Richtlinie.	Redundante Ressourcen-ARN (`arn:aws:s3:::aws-glue-*/*`) für die Aktion, die Lese-/Schreibberechtigungen für Amazon S3 Buckets gewährt, die AWS Glue verwendet, um Skripts und temporäre Dateien zu speichern. Ein Syntaxproblem wurde behoben, indem `"StringEquals"` auf `"ForAnyValue:StringLike"` geändert wurde, und die `"Effect": "Allow"` -Zeilen wurden der Zeile `"Action":` an den Stellen vorangestellt, an denen die Reihenfolge fehlerhaft war.	15. Juli 2021
AWSGlueConsoleFullAccess — Aktualisierung einer bestehenden Richtlinie.	Redundante Ressourcen-ARN (`arn:aws:s3:::aws-glue-*/*`) für die Aktion, die Lese-/Schreibberechtigungen für Amazon S3 Buckets gewährt, die AWS Glue verwendet, um Skripts und temporäre Dateien zu speichern.	15. Juli 2021
AWS Glue hat die Änderungsverfolgung gestartet.	AWS Gluehat begonnen, Änderungen an den AWS verwalteten Richtlinien zu verfolgen.	10. Juni 2021

Erteilen von Richtlinien mit dynamischem Geltungsbereich für die Auftragsausführung

AWS Glue bietet eine leistungsstarke neue Funktion: dynamische Sitzungsrichtlinien für die Ausführung von Jobs. Mit dieser Funktion können Sie benutzerdefinierte, detaillierte Berechtigungen für jeden ausgeführten Job angeben, ohne mehrere IAM-Rollen erstellen zu müssen.

Wenn Sie einen Glue-Job mithilfe der `StartJobRun` API starten, können Sie eine Inline-Sitzungsrichtlinie einbeziehen. Diese Richtlinie ändert vorübergehend die Berechtigungen der Ausführungsrolle des Jobs für die Dauer der jeweiligen Auftragsausführung. Es ist vergleichbar mit der Verwendung temporärer Anmeldeinformationen mit der `AssumeRole` API in anderen AWS Diensten.

- Verbesserte Sicherheit: Sie können die Auftragsberechtigungen auf das für jeden Lauf erforderliche Minimum beschränken.
- Vereinfachtes Management: Eliminiert die Notwendigkeit, zahlreiche IAM-Rollen für verschiedene Szenarien zu erstellen und zu verwalten.
- Flexibilität: Sie können die Berechtigungen dynamisch auf der Grundlage von Laufzeitparametern oder mandantenspezifischen Anforderungen anpassen.
- Skalierbarkeit: Diese Methode eignet sich hervorragend für Umgebungen mit mehreren Mandanten, in denen Sie Ressourcen zwischen Mandanten isolieren müssen.

Beispiele für die Gewährung der Verwendung von Richtlinien mit dynamischem Geltungsbereich:

Die folgenden Beispiele zeigen, dass Jobs Lese - und Schreibzugriff nur für einen bestimmten Amazon S3 S3-Bucket-Pfad gewährt werden, wobei der Pfad dynamisch durch die Job-Ausführungs-ID bestimmt wird. Dies veranschaulicht, wie detaillierte, ausführungsspezifische Berechtigungen für jede Auftragsausführung implementiert werden.

Von CLI

```
aws glue start-job-run \
    --job-name "your-job-name" \
    --execution-role-session-policy '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
```

```
                "Action": [
                    "s3:GetObject",
                    "s3:PutObject"
                ],
                "Resource": [
                    "arn:aws:s3:::specific-bucket/${JobRunId}/*"
                ]
            }
        ]
    }'
```

AWS GlueRessource angeben ARNs

In AWS Glue können Sie den Zugriff auf Ressourcen mithilfe einer AWS Identity and Access Management (IAM-) Richtlinie steuern. In einer Richtlinie identifizieren Sie die Ressource, für welche die Richtlinie gilt, mithilfe eines Amazon-Ressourcennamens (ARN). Nicht alle Ressourcen werden AWS Glue unterstützt ARNs.

Themen

- Datenkatalog ARNs
- ARNs für Objekte, die nicht zum Katalog gehören, in AWS Glue
- Zugriffskontrolle für nicht katalogisierte singuläre AWS Glue-API-Operationen
- Zugriffskontrolle für nichtt katalogisierte AWS Glue-API-Operationen, die mehrere Elemente abrufen
- Zugriffskontrolle für API-Operationen, die AWS Glue nicht zum Katalog gehören BatchGet

Datenkatalog ARNs

Data-Catalog-Ressourcen haben eine hierarchische Struktur, mit `catalog` als Stamm:

```
arn:aws:glue:region:account-id:catalog
```

Jedes AWS Konto hat einen einzigen Datenkatalog in einer AWS Region mit der 12-stelligen Konto-ID als Katalog-ID. Ressourcen sind eindeutig ARNs zugeordnet, wie in der folgenden Tabelle dargestellt.

Ressourcentyp	ARN-Format
Katalog	`arn:aws:glue: region:account-id :catalog` Beispiel: `arn:aws:glue:us-east-1:123456789012:catalog`
Datenbank	`arn:aws:glue: region:account-id :database/ database name` Beispiel: `arn:aws:glue:us-east-1:123456789012: database/db1`
Tabelle	`arn:aws:glue: region:account-id :table/database name/table name` Beispiel: `arn:aws:glue:us-east-1:123456789012:table/ db1/tbl1`
Katalog für föderierte S3-Tabellen (Alle Tabellen-Buckets)	`arn:aws:glue: region:account-id :catalog/s3tablescatalog` Beispiel: `arn:aws:glue:us-east-1:123456789012: catalog/s3tablescatalog`
Verbundener S3-Tabellen-Bucket-Katalog (untergeordneter Katalog)	`arn:aws:glue: region:account-id :catalog/s3tablescatalog/ bucket name` Beispiel: `arn:aws:glue:us-east-1:123456789012: catalog/s3tablescatalog/amzn-s3-demo-bucket1`
Datenbank für föderierte S3-Tabellen	`arn:aws:glue: region:account-id :database/s3tablescatalog/ child catalog name/database name` Beispiel: `arn:aws:glue:us-east-1:123456789012: database/s3tablescatalog/amzn-s3-demo-bucket1/ nsdb1`
Föderierte S3-Tabelle	`arn:aws:glue: region:account-id :table/s3tablescatalog/ child catalog name/database name/table name`

Ressourcentyp	ARN-Format
	Beispiel: `arn:aws:glue:us-east-1:123456789012:table/s3tablescatalog/amzn-s3-demo-bucket1/nsdb1/s3tbl1`
Katalog für föderierte S3-Tabellen (Ein einzelner Tabellen-Bucket, der bei Lake Formation registriert ist)	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name* Beispiel: `arn:aws:glue:us-east-1:123456789012:catalog/amzn-s3-demo-bucket1`
Datenbank für föderierte S3-Tabellen	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name*`/`*database name* Beispiel: `arn:aws:glue:us-east-1:123456789012:database/amzn-s3-demo-bucket1/nsdb1`
Föderierte S3-Tabelle	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name*`/`*database name*`/`*table name* Beispiel: `arn:aws:glue:us-east-1:123456789012:table/amzn-s3-demo-bucket1/` *nsdb1*`/s3tbl1`
Föderierter oder verwalteter Katalog (Katalog der obersten Ebene in einem Multi-Katalog)	`arn:aws:glue:` *region*`:`*account-id* `:catalog/` *top-level catalog name* Beispiel: `arn:aws:glue:us-east-1:123456789012:catalog/nscatalog` Das ARN-Format für einen verwalteten Katalog folgt derselben Struktur.
Verbundkatalog mit mehreren Ebenen (untergeordneter Katalog in einem Katalog mit mehreren Ebenen)	`arn:aws:glue:` *region*`:`*account-id* `:catalog/` *top-level catalog name*`/`*child catalog name* Beispiel: `arn:aws:glue:us-east-1:123456789012:catalog/nscatalog/dbcatalog`

Ressourcentyp	ARN-Format
Föderierte Datenbank	`arn:aws:glue:` *`region`*`:`*`account-id`* `:database/` *`name space catalog name`*`/`*`child catalog name`*`/`*`database name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:database/nscatalog/dbcatalog/schemadb`
Föderierte Tabelle	`arn:aws:glue:` *`region`*`:`*`account-id`* `:table/`*`name space catalog name`*`/`*`child catalog name`*`/`*`database name`*`/`*`table name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:table/nscatalog/dbcatalog/schemadb/rstbl1`
Link-Container für den Katalog	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog/`*`link container name`* Zum Beispiel: `arn:aws:glue:glue:us-east-1:123456789012:catalog` /linkcontainer-example
Datenbank	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog/`*`link container name`*`/`*`database name`* Zum Beispiel: /linkcontainer-example/link-db `arn:aws:glue:glue:us-east-1:123456789012:database`
Benutzerdefinierte Funktion	`arn:aws:glue:` *`region`*`:`*`account-id`* `:userDefinedFunction/` *`database name`*`/`*`user-defined function name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:userDefinedFunction/db1/func1`
Verbindung	`arn:aws:glue:` *`region`*`:`*`account-id`* `:connection/` *`connection name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:connection/connection1`

Ressourcentyp	ARN-Format
Interactive Session	`arn:aws:glue: region:account-id :session/ interactive session id` Beispiel: `arn:aws:glue:us-east-1:123456789012: session/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`

Um eine differenzierte Zugriffskontrolle zu ermöglichen, können Sie diese ARNs in Ihren IAM-Richtlinien und Ressourcenrichtlinien verwenden, um den Zugriff auf bestimmte Ressourcen zu gewähren oder zu verweigern. Platzhalter sind in den Richtlinien zulässig. Der folgende ARN stimmt beispielsweise mit allen Tabellen in der Datenbank `default` überein.

```
arn:aws:glue:us-east-1:123456789012:table/default/*
```

Important

Alle Operationen, die auf einer Data-Catalog-Ressource ausgeführt werden, benötigen die Berechtigung für die Ressource und alle Vorfahren dieser Ressource. Um beispielsweise eine Partition für eine Tabelle zu erstellen, ist eine Berechtigung für die Tabelle, die Datenbank und den Katalog erforderlich, in der bzw. dem sich die Tabelle befindet. Das folgende Beispiel zeigt die Berechtigung, die erforderlich ist, um Partitionen auf der Tabelle `PrivateTable` in der Datenbank `PrivateDatabase` im Data Catalog zu erstellen.

```
{
   "Sid": "GrantCreatePartitions",
   "Effect": "Allow",
   "Action": [
       "glue:BatchCreatePartitions"
   ],
   "Resource": [
       "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/PrivateTable",
       "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
       "arn:aws:glue:us-east-1:123456789012:catalog"
   ]
}
```

Zusätzlich zur Berechtigung für die Ressource und alle ihre Vorfahren benötigen alle Löschvorgänge die Berechtigung für alle untergeordneten Elemente dieser Ressource. Zum Beispiel erfordert das Löschen einer Datenbank die Berechtigung für alle Tabellen und benutzerdefinierten Funktionen in der Datenbank sowie für die Datenbank und den Katalog, in dem sich die Datenbank befindet. Das folgende Beispiel zeigt die Berechtigung zum Löschen der Datenbank `PrivateDatabase` im Data Catalog.

```
{
   "Sid": "GrantDeleteDatabase",
   "Effect": "Allow",
   "Action": [
       "glue:DeleteDatabase"
   ],
   "Resource": [
       "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/*",
       "arn:aws:glue:us-east-1:123456789012:userDefinedFunction/PrivateDatabase/
*",
       "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
       "arn:aws:glue:us-east-1:123456789012:catalog"
   ]
}
```

Zusammenfassend: Aktionen für Data-Catalog-Ressourcen folgen diesen Berechtigungsregeln:

- Für Katalogaktionen sind nur Berechtigungen für den Katalog erforderlich.
- Aktionen in einer Datenbank erfordern eine Berechtigung für die Datenbank und den Katalog.
- Löschaktionen in einer Datenbank erfordern eine Berechtigung für die Datenbank und den Katalog sowie alle Tabellen und benutzerdefinierten Funktionen in der Datenbank.
- Aktionen in einer Tabelle, Partition oder Tabellenversion erfordern eine Berechtigung für die Tabelle, die Datenbank und den Katalog.
- Aktionen für eine benutzerdefinierte Funktion erfordern eine Berechtigung für die benutzerdefinierte Funktion, die Datenbank und den Katalog.
- Aktionen für eine Verbindung erfordern eine Berechtigung für die Verbindung und den Katalog.

ARNs für Objekte, die nicht zum Katalog gehören, in AWS Glue

Einige AWS Glue-Ressourcen ermöglichen Zugriffsrechte auf Ressourcenebene, um den Zugriff über einen ARN zu steuern. Sie können diese ARNs in Ihren IAM-Richtlinien verwenden, um eine differenzierte Zugriffskontrolle zu ermöglichen. In der folgenden Tabelle sind die Ressourcen aufgeführt, die Ressourcen enthalten können. ARNs

Ressourcentyp	ARN-Format
Crawler	`arn:aws:glue:` *`region`*`:`*`account-id`* `:crawler/` *`crawler-name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:crawler/mycrawler`
Aufgabe	`arn:aws:glue:` *`region`*`:`*`account-id`* `:job/`*`job-name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:job/testjob`
Auslöser	`arn:aws:glue:` *`region`*`:`*`account-id`* `:trigger/` *`trigger-name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:trigger/sampletrigger`
Entwicklungsendpunkt	`arn:aws:glue:` *`region`*`:`*`account-id`* `:devEndpoint/` *`development-endpoint-name`* Beispiel: `arn:aws:glue:us-east-1:123456789012:devEndpoint/temporarydevendpoint`
Machine-Learning-Transformation	`arn:aws:glue:` *`region`*`:`*`account-id`* `:mlTransform/` *`transform-id`* Beispiel: `arn:aws:glue:us-east-1:123456789012:mlTransform/tfm-1234567890`

Zugriffskontrolle für nicht katalogisierte singuläre AWS Glue-API-Operationen

Nicht katalogisierte singuläre AWS Glue-API-Operationen wirken auf ein einzelnes Element (Entwicklungsendpunkt). Beispiele sind `GetDevEndpoint`, `CreateUpdateDevEndpoint` und `UpdateDevEndpoint`. Für diese Operationen muss eine Richtlinie den API-Namen in den Block `"action"` und den Ressourcen-ARN in den Block `"resource"` setzen.

Angenommen, Sie möchten einem Benutzer erlauben, die Operation `GetDevEndpoint` aufzurufen. Die folgende Richtlinie gewährt die minimal erforderlichen Berechtigungen für einen Endpunkt namens `myDevEndpoint-1`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MinimumPermissions",
            "Effect": "Allow",
            "Action": "glue:GetDevEndpoint",
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/
myDevEndpoint-1"
        }
    ]
}
```

Die folgende Richtlinie erlaubt `UpdateDevEndpoint` Zugriff auf Ressourcen, die mit `myDevEndpoint-` übereinstimmen, mit einem Platzhalter (*).

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionWithWildcard",
            "Effect": "Allow",
            "Action": "glue:UpdateDevEndpoint",
```

```
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/
myDevEndpoint-*"
        }
    ]
}
```

Sie können die beiden Richtlinien wie im folgenden Beispiel kombinieren. Möglicherweise wird `EntityNotFoundException` für Entwicklungsendpunkte angezeigt, deren Name mit `A` beginnt. Allerdings wird ein Zugriffsfehler zurückgegeben, wenn Sie versuchen, auf andere Entwicklungsendpunkte zuzugreifen.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CombinedPermissions",
            "Effect": "Allow",
            "Action": [
                "glue:UpdateDevEndpoint",
                "glue:GetDevEndpoint"
            ],
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/A*"
        }
    ]
}
```

Zugriffskontrolle für nichtt katalogisierte AWS Glue-API-Operationen, die mehrere Elemente abrufen

Einige AWS Glue-API-Operationen rufen mehrere Elemente ab (z. B. mehrere Entwicklungsendpunkte), z. B. `GetDevEndpoints`. Für diesen Vorgang können Sie nur eine Ressource mit Platzhalterzeichen (*) und keine spezifische Ressource angeben. ARNs

Wenn Sie beispielsweise `GetDevEndpoints` in die Richtlinie aufnehmen, muss die Ressource auf den Platzhalter (*) angewendet werden. Die einzelnen Operationen (`GetDevEndpoint`,

`CreateDevEndpoint` und `DeleteDevendpoint`) werden auch auf alle (*) Ressourcen im Beispiel bezogen.

```
{
            "Sid": "PluralAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:GetDevEndpoints",
                "glue:GetDevEndpoint",
                "glue:CreateDevEndpoint",
                "glue:UpdateDevEndpoint"
            ],
            "Resource": [
                "*"
            ]
}
```

Zugriffskontrolle für API-Operationen, die AWS Glue nicht zum Katalog gehören BatchGet

Einige AWS Glue-API-Operationen rufen mehrere Elemente ab (z. B. mehrere Entwicklungsendpunkte), z. B. `BatchGetDevEndpoints`. Für diese Operation können Sie einen ARN angeben, um den Umfang der Ressourcen einzuschränken, auf die zugegriffen werden kann.

Um beispielsweise den Zugriff auf einen bestimmten Entwicklungsendpunkt zu gewähren, fügen Sie in die Richtlinie mit dem zugehörigen Ressourcen-ARN `BatchGetDevEndpoints` ein.

```
{
            "Sid": "BatchGetAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:BatchGetDevEndpoints"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:123456789012:devEndpoint/de1"
            ]
}
```

Mit dieser Richtlinie können Sie erfolgreich auf den Entwicklungsendpunkt mit dem Namen `de1` zugreifen. Wenn Sie jedoch versuchen, auf den Entwicklungsendpunkt mit dem Namen `de2` zuzugreifen, wird ein Fehler zurückgegeben.

```
An error occurred (AccessDeniedException) when calling the BatchGetDevEndpoints
 operation: No access to any requested resource.
```

> **Important**
>
> Alternative Ansätze zum Einrichten von IAM-Richtlinien, wie z. B. die Verwendung der API-Operationen `List` und `BatchGet`, finden Sie unter Beispiele für identitätsbasierte Richtlinien für Glue AWS.

Gewährung von kontenübergreifendem Zugriff

Indem Sie kontenübergreifend Zugriff auf Data-Catalog-Ressourcen gewähren, können Ihre ETL-Aufträge zum Extrahieren, Transformieren und Laden Daten aus verschiedenen Konten abfragen und verknüpfen.

Themen

- Methoden zur Gewährung von kontenübergreifendem Zugriff in AWS Glue
- Hinzufügen oder Aktualisieren der Data-Catalog-Ressourcenrichtlinie
- Ausführen eines kontenübergreifenden API-Aufrufs
- Ausführen eines kontenübergreifenden ETL-Aufrufs
- Kontoübergreifende Protokollierung CloudTrail
- Kontoübergreifende Ressourcen-Eigentümerschaft und Buchhaltung
- Kontoübergreifende Zugriffsbeschränkungen

Methoden zur Gewährung von kontenübergreifendem Zugriff in AWS Glue

Sie können externen AWS Konten Zugriff auf Ihre Daten gewähren, indem Sie AWS Glue Methoden oder AWS Lake Formation kontenübergreifende Zuschüsse verwenden. Die AWS Glue Methoden verwenden AWS Identity and Access Management (IAM-) Richtlinien, um eine differenzierte Zugriffskontrolle zu erreichen. Lake Formation verwendet ein einfacheres `GRANT/REVOKE`-Berechtigungsmodell ähnlich den `GRANT/REVOKE`-Befehlen in einem relationalen Datenbanksystem.

In diesem Abschnitt wird die Verwendung der AWS Glue-Methoden beschrieben. Weitere Informationen zur kontoübergreifenden Verwendung von Lake Formation finden Sie unter Erteilen von Lake-Formation-Berechtigungen im AWS Lake Formation -Entwicklerhandbuch.

Es gibt zwei AWS Glue-Methoden zum Erteilen von kontenübergreifendem Zugriff auf eine Ressource:

- Verwenden einer Data-Catalog-Ressourcenrichtlinie
- Verwenden einer IAM-Rolle

Eine Ressourcenrichtlinie verwenden, um kontenübergreifenden Zugriff zu gewähren

Im Folgenden werden die allgemeinen Schritte zum Erteilen des kontenübergreifenden Zugriffs mithilfe einer Data-Catalog-Ressourcenrichtlinie beschrieben:

1. Ein Administrator (oder eine andere autorisierte Identität) in Konto A fügt eine Ressourcenrichtlinie an den Data Catalog in Konto A an. Diese Richtlinie gewährt Konto B spezifische kontenübergreifende Berechtigungen, um Operationen mit einer Ressource im Katalog von Konto A durchzuführen.
2. Ein Administrator in Konto B fügt eine IAM-Richtlinie für eine IAM-Identität in Konto B an, die die von Konto A erhaltenen Berechtigungen delegiert.

   Die Identität in Konto B hat nun Zugriff auf die angegebene Ressource in Konto A.

   Die Identität benötigt die Berechtigung sowohl vom Ressourcenbesitzer (Konto A) als auch von ihrem übergeordneten Konto (Konto B), um auf die Ressource zugreifen zu können.

Gewährung von kontenübergreifendem Zugriff mithilfe einer IAM-Rolle

Im Folgenden werden die allgemeinen Schritte zum Erteilen des kontenübergreifenden Zugriffs mithilfe einer IAM-Rolle beschrieben:

1. Ein Administrator (oder eine andere autorisierte Identität) in dem Konto, das die Ressource besitzt (Konto A), erstellt eine IAM-Rolle.
2. Der Administrator in Konto A fügt eine Richtlinie an die Rolle an, die kontenübergreifende Berechtigungen für den Zugriff auf die betreffende Ressource gewährt.
3. Der Administrator in Konto A fügt eine Vertrauensrichtlinie an die Rolle an, die eine IAM-Identität in einem anderen Konto (Konto B) als Prinzipal angibt, der die Rolle übernehmen kann.

Der Principal in der Vertrauensrichtlinie kann auch ein AWS Dienstprinzipal sein, wenn Sie eine AWS Serviceberechtigung zur Übernahme der Rolle erteilen möchten.

4. Ein Administrator in Konto B delegiert nun Berechtigungen an eine oder mehrere IAM-Identitäten in Konto B, damit sie diese Rolle übernehmen können. Auf diese Weise erhalten diese Identitäten in Konto B Zugriff auf die Ressource in Konto A.

Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter Zugriffsverwaltung im IAM-Benutzerhandbuch. Weitere Informationen zu Benutzern, Gruppen, Rollen und Berechtigungen finden Sie unter Identitäten (Benutzer, Gruppen und Rollen) im IAM-Benutzerhandbuch.

Einen Vergleich dieser beiden Ansätze finden Sie unter Wie sich IAM-Rollen und ressourcenbasierten Richtlinien unterscheiden im IAM-Benutzerhandbuch. AWS Glue unterstützt beide Optionen, mit der Einschränkung, dass eine Ressourcenrichtlinie nur Zugriff auf Data-Catalog-Ressourcen gewähren kann.

Um zum Beispiel der `Dev`-Rolle in Konto B Zugriff auf die Datenbank `db1` in Konto A zu geben, fügen Sie die folgende Ressourcenrichtlinie an den Katalog in Konto A an.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase"
            ],
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:role/Dev"
                ]
            },
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/db1"
            ]
        }
```

```
    ]
}
```

Darüber hinaus müsste Konto B folgende IAM-Richtlinie zur `Dev`-Rolle anfügen, bevor sie tatsächlich Zugriff auf `db1` in Konto A erhält.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/db1"
            ]
        }
    ]
}
```

Hinzufügen oder Aktualisieren der Data-Catalog-Ressourcenrichtlinie

Sie können die AWS Glue Datenkatalog-Ressourcenrichtlinie mithilfe der Konsole, der API oder AWS Command Line Interface (AWS CLI) hinzufügen oder aktualisieren.

> **Important**
>
> Wenn Sie bereits kontoübergreifende Berechtigungen von Ihrem Konto mit AWS Lake Formation erstellt haben, erfordert das Hinzufügen oder Aktualisieren der Data-Catalog-Ressourcenrichtlinie einen zusätzlichen Schritt. Weitere Informationen finden Sie unter Verwalten von kontenübergreifenden Berechtigungen mithilfe von AWS Glue und Lake Formation im AWS Lake Formation -Entwicklerhandbuch.
> Um festzustellen, ob kontenübergreifende Berechtigungen für Lake Formation vorhanden sind, verwenden Sie den `glue:GetResourcePolicies`-API-Vorgang oder die AWS CLI.

Wenn `glue:GetResourcePolicies` andere Richtlinien als eine bereits vorhandene Data-Catalog-Richtlinie zurückgibt, sind Lake-Formation-Berechtigungen vorhanden. Weitere Informationen finden Sie im AWS Lake Formation Entwicklerhandbuch unter Alle kontoübergreifenden Zuschüsse mithilfe des GetResourcePolicies API-Vorgangs anzeigen.

Data-Catalog-Ressourcenrichtlinie hinzufügen oder aktualisieren (Konsole)

1. Öffnen Sie die AWS Glue-Konsole unter https://console.aws.amazon.com/glue/.

   Melden Sie sich als AWS Identity and Access Management (IAM-) Administratorbenutzer an, der über die `glue:PutResourcePolicy` entsprechende Berechtigung verfügt.
2. Wählen Sie im Navigationsbereich Settings (Einstellungen).
3. Fügen Sie auf der Seite Data catalog settings (Data-Catalog-Einstellungen) unter Permissions (Berechtigungen) eine Ressourcenrichtlinie in den Textbereich ein. Wählen Sie dann Speichern.

   Wenn die Konsole eine Warnmeldung anzeigt, dass die Berechtigungen in der Richtlinie zusätzlich zu den mit Lake Formation erteilten Berechtigungen sind, wählen Sie Fortfahren.

Data-Catalog-Ressourcenrichtlinie hinzufügen oder aktualisieren (AWS CLI)

- Senden Sie einen `aws glue put-resource-policy`-Befehl. Wenn bereits Berechtigungen für Lake Formation vorhanden sind, stellen Sie sicher, dass Sie die Option `--enable-hybrid` mit dem Wert `'TRUE'` auswählen.

  Beispiele für die Verwendung dieses Befehls finden Sie unter Beispiele für ressourcenbasierte Richtlinien für Glue AWS.

Ausführen eines kontenübergreifenden API-Aufrufs

Alle AWS Glue Data Catalog-Operationen haben ein Feld `CatalogId`. Wenn die erforderlichen Berechtigungen zugewiesen wurden, um den kontenübergreifenden Zugriff zu ermöglichen, kann ein Aufrufer Data-Catalog-API-Aufrufe über mehrere Konten hinweg ausführen. Hierfügt übergibt er die ID des AWS -Zielkontos in `CatalogId`, um auf die Ressource in diesem Zielkonto zuzugreifen.

Wenn kein `CatalogId`-Wert angegeben wird, verwendet AWS Glue standardmäßig die eigene Konto-ID des Aufrufers, und der Aufruf ist nicht kontoübergreifend.

Ausführen eines kontenübergreifenden ETL-Aufrufs

Einige AWS Glue PySpark und Scala APIs haben ein Katalog-ID-Feld. Wenn alle erforderlichen Berechtigungen erteilt wurden, um den kontenübergreifenden Zugriff zu ermöglichen, kann ein ETL-Job API-Operationen für alle Konten ausführen PySpark und Scala aufrufen, indem die AWS Zielkonto-ID im Feld „Katalog-ID“ übergeben wird, um auf Datenkatalogressourcen in einem Zielkonto zuzugreifen.

Wenn kein Katalog-ID-Wert angegeben ist, verwendet AWS Glue standardmäßig die eigene Konto-ID des Aufrufers, und der Aufruf ist nicht kontoübergreifend.

Informationen zu PySpark APIs dieser Unterstützung `catalog_id` finden Sie unter. GlueContext Klasse Informationen zu APIs dieser Unterstützung durch Scala finden Sie `catalogId` unterAWS Glue Scala GlueContext APIs.

Das folgende Beispiel zeigt die Berechtigungen, die der Berechtigungsempfänger benötigt, um einen ETL-Auftrag auszuführen. In diesem Beispiel *`grantee-account-id`* ist der Client, `catalog-id` der den Job ausführt, und er *`grantor-account-id`* ist der Besitzer der Ressource. In diesem Beispiel wird die Berechtigung für alle Katalogressourcen im Konto des Eigentümers gewährt. Um den Umfang der gewährten Ressourcen einzuschränken, können Sie spezifische Ressourcen ARNs für den Katalog, die Datenbank, die Tabelle und die Verbindung angeben.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetConnection",
                "glue:GetDatabase",
                "glue:GetTable",
                "glue:GetPartition"
            ],
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
```

```
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:*"
            ]
        }
    ]
}
```

> **Note**
>
> Wenn eine Tabelle im Konto des Eigentümers auf einen Amazon-S3-Speicherort verweist, der sich auch im Konto des Eigentümers befindet, muss die IAM-Rolle, mit der ein ETL-Auftrag im Konto des Berechtigungsempfängers ausgeführt wird, die Berechtigung haben, Objekte aufzulisten und aus dem Konto des Eigentümers abzurufen.

Da der Client in Konto A bereits die Berechtigung zum Erstellen und Ausführen von ETL-Aufträgen hat, sind die grundlegenden Schritte zum Einrichten eines ETL-Auftrags für den kontoübergreifenden Zugriff die folgenden:

1. Kontoübergreifenden Datenzugriff erlauben (überspringen Sie diesen Schritt, wenn der kontoübergreifende Zugriff in Amazon S3 bereits eingerichtet ist).
   a. Aktualisieren Sie die Amazon-S3-Bucket-Richtlinie in Konto B, um den kontoübergreifenden Zugriff von Konto A aus zu ermöglichen.
   b. Aktualisieren Sie die IAM-Richtlinie in Konto A, um den Zugriff auf den Bucket in Konto B zu ermöglichen.
2. Kontoübergreifenden Zugriff auf den Data Catalog gewähren
   a. Erstellen oder aktualisieren Sie die Ressourcenrichtlinie, die an den Data Catalog in Konto B angefügt ist, um den Zugriff von Konto A aus zu ermöglichen.
   b. Aktualisieren Sie die IAM-Richtlinie in Konto A, um den Zugriff auf den Data Catalog in Konto B zu ermöglichen.

Kontoübergreifende Protokollierung CloudTrail

Wenn ein ETL-Job (AWS GlueExtrahieren, Transformieren und Laden) auf die zugrunde liegenden Daten einer Datenkatalogtabelle zugreift, die über AWS Lake Formation kontenübergreifende

Zuschüsse gemeinsam genutzt wird, kommt es zu einem zusätzlichen AWS CloudTrail Protokollierungsverhalten.

Für die Zwecke dieser Erläuterung ist das AWS Konto, das die Tabelle gemeinsam genutzt hat, das Besitzerkonto und das Konto, mit dem die Tabelle geteilt wurde, das Empfängerkonto. Wenn ein ETL-Auftrag im Empfängerkonto auf Daten in der Tabelle im Besitzerkonto zugreift, wird das CloudTrail Datenzugriffsereignis, das den Protokollen für das Empfängerkonto hinzugefügt wird, in die Protokolle des Besitzerkontos CloudTrail kopiert. Auf diese Weise können Besitzerkonten Datenzugriffe durch verschiedene Empfängerkonten verfolgen. Standardmäßig enthalten die CloudTrail Ereignisse keinen für Menschen lesbaren Principal Identifier (Principal-ARN). Ein Administrator des Empfängerkontos kann sich dafür entscheiden, den Principal ARN in die Protokolle aufzunehmen.

Weitere Informationen finden Sie unter Kontoübergreifende CloudTrail Protokollierung im Entwicklerhandbuch.AWS Lake Formation

Weitere Informationen finden Sie unter:

- the section called “Protokollierung und Überwachung”

Kontoübergreifende Ressourcen-Eigentümerschaft und Buchhaltung

Wenn ein Benutzer in einem AWS Konto (Konto A) eine neue Ressource wie eine Datenbank in einem anderen Konto (Konto B) erstellt, gehört diese Ressource dann Konto B, dem Konto, in dem sie erstellt wurde. Ein Administrator in Konto B erhält automatisch volle Berechtigungen für den Zugriff auf die neue Ressource, einschließlich Lesen, Schreiben und Erteilen von Zugriffsrechten für ein drittes Konto. Der Benutzer in Konto A kann nur dann auf die Ressource zugreifen, die er gerade erstellt hat, wenn er die entsprechenden Berechtigungen hat, die von Konto B gewährt werden.

Speicherkosten und andere Kosten, die direkt mit der neuen Ressource verbunden sind, werden dem Konto B, dem Ressourcenbesitzer, in Rechnung gestellt. Die Kosten für Anfragen des Benutzers, der die Ressource angelegt hat, werden dem Konto des Antragstellers, Konto A, in Rechnung gestellt.

Weitere Informationen zur AWS Glue Abrechnung und Preisgestaltung finden Sie unter So funktioniert die AWS Preisgestaltung.

Kontoübergreifende Zugriffsbeschränkungen

Kontoübergreifender AWS Glue-Zugriff hat folgende Einschränkungen:

- Der kontoübergreifende Zugriff auf AWS Glue ist nicht zulässig, wenn Sie Datenbanken und Tabellen mit Amazon Athena oder Amazon Redshift Spectrum erstellt haben, bevor eine Region AWS Glue unterstützt hat, und das Konto des Ressourceneigentümers den Amazon Athena-Data Catalog nicht nach AWS Glue migriert hat. Den aktuellen Migrationsstatus erfahren Sie über GetCatalogImportStatus (get_catalog_import_status). Weitere Informationen zur Migration eines Athena-Katalogs zu AWS Glue finden Sie unter Upgrade auf den AWS Glue Data Catalog step-by-step im Amazon Athena Athena-Benutzerhandbuch.
- Der kontoübergreifende Zugriff wird nur für Data-Catalog-Ressourcen unterstützt, einschließlich Datenbanken, Tabellen, benutzerdefinierten Funktionen und Verbindungen.
- Der kontoübergreifende Zugriff auf den Data Catalog von Athena erfordert, dass Sie den Katalog als eine Athena `DataCatalog`-Ressource registrieren. Detaillierte Anweisungen finden Sie unter Registrierung eines AWS Glue Data Catalog von einem anderen Konto im Amazon Athena-Benutzerhandbuch.

Fehlerbehebung bei Identität und Zugriff auf AWS Glue

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit AWS Glue und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion in AWS Glue durchzuführen
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine AWS Glue-Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion in AWS Glue durchzuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `glue:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 glue:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `glue:GetWidget`-Aktion auf die `my-example-widget`-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an AWS Glue übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in AWS Glue auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine AWS Glue-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob AWS Glue diese Funktionen unterstützt, finden Sie unterSo funktioniert AWS Glue mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

AWS Lake Formation Modelle für die Zugriffskontrolle

AWS Glue 5.0 unterstützt zwei Modelle für den Datenzugriff über AWS Lake Formation:

Themen

- Verwenden von AWS Glue with AWS Lake Formation für den vollständigen Tabellenzugriff
- Verwendung von AWS Glue mit AWS Lake Formation für eine feinkörnige Zugriffskontrolle

Verwenden von AWS Glue with AWS Lake Formation für den vollständigen Tabellenzugriff

Einführung in den vollständigen Tabellenzugriff

AWS Glue 5.0 unterstützt die Steuerung des vollständigen Tabellenzugriffs (FTA) in Apache Spark auf der Grundlage Ihrer in AWS Lake Formation definierten Richtlinien. Diese Funktion ermöglicht Lese- und Schreiboperationen von Ihren AWS Glue Spark-Jobs aus auf AWS Lake Formation registrierten Tabellen, wenn die Jobrolle vollen Tabellenzugriff hat. FTA ist ideal für Anwendungsfälle, bei denen Sicherheitsvorschriften auf Tabellenebene eingehalten werden müssen, und unterstützt Spark-Funktionen wie Resilient Distributed Datasets (RDDs), benutzerdefinierte Bibliotheken und benutzerdefinierte Funktionen (UDFs) mit AWS Lake Formation Tabellen.

Wenn ein AWS Glue Spark-Job für Full Table Access (FTA) konfiguriert ist, werden AWS Lake Formation Anmeldeinformationen für read/write Amazon S3 S3-Daten für AWS Lake Formation registrierte Tabellen verwendet, während die Runtime-Rollen-Anmeldeinformationen des Jobs für read/write Tabellen verwendet werden, bei AWS Lake Formation denen nicht registriert ist. Diese Funktion ermöglicht DML-Operationen (Data Manipulation Language) wie CREATE-, ALTER-, DELETE-, UPDATE- und MERGE INTO-Anweisungen für Apache Hive- und Iceberg-Tabellen.

> **Note**
>
> Überprüfen Sie Ihre Anforderungen und stellen Sie fest, ob Fine-Grained Access Control (FGAC) oder Full Table Access (FTA) Ihren Anforderungen entspricht. Für einen bestimmten Job kann nur eine AWS Lake Formation Berechtigungsmethode aktiviert werden. AWS Glue Ein Job kann nicht gleichzeitig Full Table Access (FTA) und Fine-Grained Access Control (FGAC) gleichzeitig ausführen.

So funktioniert Full-Table Access (FTA) auf AWS Glue

AWS Lake Formation bietet zwei Ansätze für die Datenzugriffskontrolle: Fine-Grained Access Control (FGAC) und Full Table Access (FTA). FGAC bietet verbesserte Sicherheit durch Filterung auf Spalten-, Zeilen- und Zellenebene und eignet sich ideal für Szenarien, in denen detaillierte Berechtigungen erforderlich sind. FTA ist ideal für einfache Zugriffskontrollszenarien, in denen Sie Berechtigungen auf Tabellenebene benötigen. Es vereinfacht die Implementierung, da kein detaillierter Zugriffsmodus erforderlich ist, verbessert die Leistung und senkt die Kosten, da Systemtreiber und Systemausführungen vermieden werden, und unterstützt sowohl Lese- als auch Schreibvorgänge (einschließlich der Befehle CREATE, ALTER, DELETE, UPDATE und MERGE INTO).

In AWS Glue 4.0 funktionierte der AWS Lake Formation basierte Datenzugriff über die Klasse, die GlueContext Dienstprogrammklasse, die von bereitgestellt wird. AWS Glue In AWS Glue 5.0 ist der AWS Lake Formation basierte Datenzugriff über systemeigenes Spark-SQL, Spark DataFrames, verfügbar und wird weiterhin über GlueContext Klassen unterstützt.

Implementierung des vollständigen Tabellenzugriffs

Schritt 1: Aktivieren Sie den vollständigen Tabellenzugriff in AWS Lake Formation

Um den Modus Full Table Access (FTA) zu verwenden, müssen Sie Abfrage-Engines von Drittanbietern den Zugriff auf Daten ermöglichen, ohne dass die IAM-Sitzung-Tag-Validierung aktiviert

ist. AWS Lake Formation Folgen Sie zur Aktivierung den Schritten unter Anwendungsintegration für vollständigen Tabellenzugriff.

Schritt 2: Richten Sie die IAM-Berechtigungen für die Job-Runtime-Rolle ein

Für den Lese- oder Schreibzugriff auf die zugrunde liegenden Daten benötigt eine Job-Runtime-Rolle zusätzlich zu den AWS Lake Formation Berechtigungen die `lakeformation:GetDataAccess` IAM-Berechtigung. Mit dieser Berechtigung wird der Anforderung temporärer Anmeldeinformationen für den Zugriff auf die Daten stattgegeben. AWS Lake Formation

Im Folgenden finden Sie eine Beispielrichtlinie für die Bereitstellung von IAM-Berechtigungen für den Zugriff auf ein Skript in Amazon S3, für das Hochladen von Protokollen auf Amazon S3, AWS Glue API-Berechtigungen und Zugriffsberechtigungen. AWS Lake Formation

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ScriptAccess",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/scripts/*"
      ]
    },
    {
      "Sid": "LoggingAccess",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/logs/*"
      ]
    },
    {
      "Sid": "GlueCatalogAccess",
      "Effect": "Allow",
```

```
      "Action": [
        "glue:GetDatabase",
        "glue:GetDatabases",
        "glue:GetTable",
        "glue:GetTables",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:CreateTable",
        "glue:UpdateTable"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:123456789012:catalog",
        "arn:aws:glue:us-east-1:123456789012:database/default",
        "arn:aws:glue:us-east-1:123456789012:table/default/*"
      ]
    },
    {
      "Sid": "LakeFormationAccess",
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": "*"
    }
  ]
}
```

Schritt 2.1 Berechtigungen konfigurieren AWS Lake Formation

AWS Glue Spark-Jobs, die Daten aus Amazon S3 lesen, benötigen die AWS Lake Formation SELECT-Berechtigung.

AWS Glue Spark-Jobs, für die write/delete Daten in Amazon S3 ist eine AWS Lake Formation ALL-Genehmigung erforderlich.

AWS Glue Spark-Jobs, die mit dem AWS Glue Datenkatalog interagieren, benötigen die entsprechenden DESCRIBE-, ALTER- und DROP-Berechtigungen.

Schritt 3: Initialisieren Sie eine Spark-Sitzung für den vollständigen Tabellenzugriff mit AWS Lake Formation

Um auf Tabellen zuzugreifen AWS Lake Formation, bei denen registriert ist, müssen bei der Spark-Initialisierung die folgenden Konfigurationen festgelegt werden, um Spark für die Verwendung AWS Lake Formation von Anmeldeinformationen zu konfigurieren.

Um auf Tabellen zuzugreifen AWS Lake Formation, mit denen Sie registriert sind, müssen Sie Ihre Spark-Sitzung explizit für die Verwendung von AWS Lake Formation Anmeldeinformationen konfigurieren. Fügen Sie bei der Initialisierung Ihrer Spark-Sitzung die folgenden Konfigurationen hinzu:

```
from pyspark.sql import SparkSession

# Initialize Spark session with Lake Formation configurations
spark = SparkSession.builder \
    .appName("Lake Formation Full Table Access") \
    .config("spark.sql.catalog.glue_catalog",
 "org.apache.spark.sql.catalog.hive.GlueCatalog") \
    .config("spark.sql.catalog.glue_catalog.glue.lakeformation-enabled", "true") \
    .config("spark.sql.defaultCatalog", "glue_catalog") \
    .getOrCreate()
```

Die wichtigsten Konfigurationen:

- `spark.sql.catalog.glue_catalog`: Registriert einen Katalog mit dem Namen „glue_catalog“, der die Implementierung verwendet GlueCatalog
- `spark.sql.catalog.glue_catalog.glue.lakeformation-enabled`: Aktiviert explizit die AWS Lake Formation Integration für diesen Katalog
- Der Katalogname (in diesem Beispiel „glue_catalog“) kann angepasst werden, muss aber in beiden Konfigurationseinstellungen konsistent sein

Hive

```
--conf spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS
 Glue.accesscontrol.AWSLakeFormationCredentialResolver
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
```

```
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

Iceberg

```
--conf spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS
 Glue.accesscontrol.AWSLakeFormationCredentialResolver
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
--conf spark.sql.catalog.<catalog>.AWS Glue.lakeformation-enabled=true
```

- `spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS Glue.accesscontrol.AWSLakeFormationCredentialResolver`: Konfigurieren Sie das EMR-Dateisystem (EMRFS) so, dass AWS Lake Formation S3-Anmeldeinformationen für registrierte Tabellen verwendet werden. AWS Lake Formation Wenn die Tabelle nicht registriert ist, verwenden Sie die Anmeldeinformationen für die Runtime-Rolle des Jobs.
- `spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=trueUndspark.hadoop.fs.s3.` Konfigurieren Sie EMRFS so, dass beim Erstellen von S3-Ordnern der Inhaltstypheader application/x-directory anstelle des Suffixes $folder$ verwendet wird. Dies ist beim Lesen von AWS Lake Formation Tabellen erforderlich, da AWS Lake Formation Anmeldeinformationen das Lesen von Tabellenordnern mit dem Suffix $folder$ nicht zulassen.
- `spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true`: Konfigurieren Sie Spark so, dass die Überprüfung, ob der Tabellenspeicherort leer ist, vor der Erstellung übersprungen wird. Dies ist für AWS Lake Formation registrierte Tabellen erforderlich, da AWS Lake Formation Anmeldeinformationen zur Überprüfung des leeren Speicherorts erst nach der Erstellung der AWS Glue Datenkatalogtabelle verfügbar sind. Ohne diese Konfiguration validieren die Anmeldeinformationen für die Runtime-Rolle des Jobs den Speicherort der leeren Tabelle.
- `spark.sql.catalog.createDirectoryAfterTable.enabled=true`: Konfigurieren Sie Spark so, dass der Amazon S3 S3-Ordner nach der Tabellenerstellung im Hive-Metastore erstellt wird. Dies ist für AWS Lake Formation registrierte Tabellen erforderlich, da AWS Lake Formation

Anmeldeinformationen für die Erstellung des Amazon S3 S3-Ordners erst nach der Erstellung der AWS Glue Datenkatalogtabelle verfügbar sind.

- `spark.sql.catalog.dropDirectoryBeforeTable.enabled=true`: Konfigurieren Sie Spark so, dass der Amazon S3 S3-Ordner vor dem Löschen der Tabelle im Hive-Metastore gelöscht wird. Dies ist für AWS Lake Formation registrierte Tabellen erforderlich, da die AWS Lake Formation Anmeldeinformationen zum Löschen des S3-Ordners nach dem Löschen der Tabelle aus dem AWS Glue Datenkatalog nicht verfügbar sind.
- `spark.sql.catalog.<catalog>.AWS Glue.lakeformation-enabled=true`: Konfigurieren Sie den Iceberg-Katalog so, dass er AWS Lake Formation Amazon S3 S3-Anmeldeinformationen für AWS Lake Formation registrierte Tabellen verwendet. Wenn die Tabelle nicht registriert ist, verwenden Sie Standardanmeldedaten für die Umgebung.

Nutzungsmuster

Verwenden von FTA mit DataFrames

Für Benutzer, die mit Spark vertraut sind, DataFrames kann es mit AWS Lake Formation Full Table Access verwendet werden.

AWS Glue 5.0 bietet native Spark-Unterstützung für Lake Formation Full Table Access und vereinfacht so die Arbeit mit geschützten Tabellen. Diese Funktion ermöglicht es AWS Glue AWS Glue 5.0-Spark-Jobs, Daten direkt zu lesen und zu schreiben, wenn vollständiger Tabellenzugriff gewährt wird, wodurch Einschränkungen aufgehoben werden, die zuvor bestimmte ETL-Operationen (Extrahieren, Transformieren und Laden) einschränkten. Sie können jetzt erweiterte Spark-Funktionen wie Resilient Distributed Datasets (RDDs), benutzerdefinierte Bibliotheken und benutzerdefinierte Funktionen (UDFs) mit AWS Lake Formation Tabellen nutzen.

Natives Spark-FTA in 5.0 AWS Glue

AWS Glue 5.0 unterstützt die Steuerung des vollständigen Tabellenzugriffs (FTA) in Apache Spark auf der Grundlage Ihrer in definierten Richtlinien. AWS Lake Formation Diese Kontrollebene ist ideal für Anwendungsfälle, bei denen Sicherheitsvorschriften auf Tabellenebene eingehalten werden müssen.

Beispiel für eine Apache Iceberg-Tabelle

```
from pyspark.sql import SparkSession

catalog_name = "spark_catalog"
```

```
aws_region = "us-east-1"
aws_account_id = "123456789012"
warehouse_path = "s3://amzn-s3-demo-bucket/warehouse/"

spark = SparkSession.builder \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkSessionCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.warehouse", f"{warehouse_path}") \
    .config(f"spark.sql.catalog.{catalog_name}.client.region",f"{aws_region}") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.account-id",f"{aws_account_id}") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled","true") \
    .config(f"spark.sql.catalog.dropDirectoryBeforeTable.enabled", "true") \
    .config(f"spark.sql.catalog.{catalog_name}.catalog-impl",
 "org.apache.iceberg.aws.glue.GlueCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.io-impl",
 "org.apache.iceberg.aws.s3.S3FileIO") \
    .config("spark.sql.defaultCatalog", catalog_name) \  # Add this line
    .getOrCreate()

database_name = "your_database"
table_name = "your_table"

df = spark.sql(f"select * from {database_name}.{table_name}")
df.show()
```

Erforderliche IAM-Berechtigungen

Ihre Rolle bei der AWS Glue Auftragsausführung muss Folgendes haben:

```
{
    "Action": "lakeformation:GetDataAccess",
    "Resource": "*",
    "Effect": "Allow"
}
```

Außerdem die entsprechenden S3-Zugriffsberechtigungen für Ihre Datenstandorte.

Konfiguration der Lake Formation

Vor der Verwendung von nativem Spark FTA in AWS Glue 5.0:

1. Erlauben Sie Query-Engines von Drittanbietern den Zugriff auf Daten ohne Überprüfung des IAM-Sitzungs-Tags in AWS Lake Formation
2. Erteilen Sie Ihrer AWS Glue Jobausführungsrolle über AWS Lake Formation die Konsole die entsprechenden Tabellenberechtigungen
3. Konfigurieren Sie Ihre Spark-Sitzung mit den erforderlichen Parametern, die im obigen Beispiel gezeigt werden

Verwenden Sie FTA mit DynamicFrames

AWS Glue Die native Version DynamicFrames kann mit AWS Lake Formation vollständigem Tabellenzugriff für optimierte ETL-Operationen verwendet werden. Full Table Access (FTA) bietet ein Sicherheitsmodell, das Berechtigungen auf Tabellenebene gewährt. Dies ermöglicht im Vergleich zu Fine-Grained Access Control (FGAC) eine schnellere Datenverarbeitung, da der Aufwand für Berechtigungsprüfungen auf Zeilen- und Spaltenebene umgangen wird. Dieser Ansatz ist nützlich, wenn Sie ganze Tabellen verarbeiten müssen und Berechtigungen auf Tabellenebene Ihren Sicherheitsanforderungen entsprechen.

In AWS Glue Version 4.0 war DynamicFrames mit FTA eine spezifische Konfiguration erforderlich. GlueContext Bestehender AWS Glue DynamicFrame 4.0-Code mit FTA wird in AWS Glue 5.0 weiterhin funktionieren, aber die neuere Version bietet auch native Spark FTA-Unterstützung mit größerer Flexibilität. Für neue Entwicklungen sollten Sie den in DataFrames diesem Abschnitt beschriebenen systemeigenen Spark-Ansatz in Betracht ziehen, insbesondere wenn Sie zusätzliche Funktionen wie Resilient Distributed Datasets (RDDs), benutzerdefinierte Bibliotheken und benutzerdefinierte Funktionen (UDFs) mit Tabellen benötigen. AWS Lake Formation

Erforderliche Berechtigungen

Die IAM-Rolle, die Ihren Glue-Job ausführt, muss Folgendes haben:

- `lakeformation:GetDataAccess`-Berechtigung
- Entsprechende Lake Formation-Tabellenberechtigungen, die über die Lake Formation Formation-Konsole erteilt wurden

Beispiel für eine DynamicFrame Implementierung in AWS Glue 5.0

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext
```

```
# Initialize Glue context
sc = SparkContext()
glueContext = GlueContext(sc)

# Configure catalog for Iceberg tables
catalog_name = "glue_catalog"
aws_region = "us-east-1"
aws_account_id = "123456789012"
warehouse_path = "s3://amzn-s3-demo-bucket/warehouse/"

spark = glueContext.spark_session
spark.conf.set(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.warehouse", f"{warehouse_path}")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.catalog-impl",
 "org.apache.iceberg.aws.glue.GlueCatalog")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.io-impl",
 "org.apache.iceberg.aws.s3.S3FileIO")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled","true")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.client.region",f"{aws_region}")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.glue.id", f"{aws_account_id}")

# Read Lake Formation-protected table with DynamicFrame
df = glueContext.create_data_frame.from_catalog(
    database="your_database",
    table_name="your_table"
)
```

Zusätzliche Konfiguration

Konfigurieren Sie den vollständigen Tabellenzugriffsmodus in AWS Glue Studio Notebooks

Um über interaktive Spark-Sitzungen in AWS Glue Studio Notebooks auf AWS Lake Formation registrierte Tabellen zuzugreifen, müssen Sie den Kompatibilitätsberechtigungsmodus verwenden. Verwenden Sie den `%%configure` magischen Befehl, um Ihre Spark-Konfiguration einzurichten, bevor Sie Ihre interaktive Sitzung starten. Diese Konfiguration muss der erste Befehl in Ihrem Notizbuch sein, da sie nach dem Start der Sitzung nicht mehr angewendet werden kann. Wählen Sie die Konfiguration basierend auf Ihrem Tabellentyp aus:

Für Hive-Tabellen

```
%%configure
```

```
--conf
 spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.glue.accesscontrol.AWSLakeFormationC
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

Für Iceberg-Tische

```
%%configure
--conf
 spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.glue.accesscontrol.AWSLakeFormationC
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
--conf spark.sql.catalog.glue_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.glue_catalog.warehouse=s3://example-s3-bucket_DATA_LOCATION
--conf spark.sql.catalog.glue_catalog.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog
--conf spark.sql.catalog.glue_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO
--conf spark.sql.catalog.glue_catalog.glue.account-id=ACCOUNT_ID
--conf spark.sql.catalog.glue_catalog.glue.region=REGION
```

Ersetzen Sie die Platzhalter:

- S3_DATA_LOCATION: *s3://amzn-s3-demo-bucket*
- REGION: *AWS Region (e.g., us-east-1)*
- KONTO-ID: *Your AWS Account ID*

> **Note**
>
> Sie müssen diese Konfigurationen festlegen, bevor Sie Spark-Operationen in Ihrem Notebook ausführen.

Unterstützte Vorgänge

Bei diesen Vorgängen werden AWS Lake Formation Anmeldeinformationen für den Zugriff auf die Tabellendaten verwendet.

> **Note**
>
> Bei der Aktivierung AWS Lake Formation:
>
> - Für FTA: Aktivieren Sie die Spark-Konfiguration `spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled`

- CREATE TABLE
- ALTER TABLE
- INSERT INTO
- INSERT OVERWRITE
- SELECT
- UPDATE
- MERGE INTO
- DELETE FROM
- TABELLE ANALYSIEREN
- TABELLE REPARIEREN
- DROP TABLE
- Spark-Datenquellenabfragen
- Spark-Datenquellenschreibvorgänge

> **Note**
>
> Operationen, die oben nicht aufgeführt sind, verwenden weiterhin IAM-Berechtigungen für den Zugriff auf Tabellendaten.

Migration von AWS Glue 4.0 auf 5.0 FTA AWS Glue

Bei der Migration von AWS Glue 4.0 GlueContext FTA auf AWS Glue 5.0 native Spark FTA:

1. Erlauben Sie Abfrage-Engines von Drittanbietern den Zugriff auf Daten, ohne dass die Überprüfung des IAM-Sitzungs-Tags aktiviert ist. AWS Lake Formation FolgenSchritt 1: Aktivieren Sie den vollständigen Tabellenzugriff in AWS Lake Formation.
2. Sie müssen die Job-Runtime-Rolle nicht ändern. Stellen Sie jedoch sicher, dass die Rolle „ AWS Glue Job Execution“ über die Berechtigung Lakeformation: GetDataAccess IAM verfügt.
3. Ändern Sie die Spark-Sitzungskonfigurationen im Skript. Stellen Sie sicher, dass die folgenden Spark-Konfigurationen vorhanden sind:

```
--conf spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
--conf spark.sql.catalog.spark_catalog.warehouse=s3://<bucket-name>/warehouse/
--conf spark.sql.catalog.spark_catalog.client.region=<REGION>
--conf spark.sql.catalog.spark_catalog.glue.account-id=ACCOUNT_ID
--conf spark.sql.catalog.spark_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

4. Aktualisieren Sie das Skript so, dass GlueContext DataFrames es auf natives Spark umgestellt wird DataFrames.
5. Aktualisieren Sie Ihren AWS Glue Job, um AWS Glue 5.0 zu verwenden

Überlegungen und Einschränkungen

- Wenn eine Hive-Tabelle mit einem Job erstellt wird, für den der vollständige Tabellenzugriff nicht aktiviert ist, und keine Datensätze eingefügt werden, schlagen nachfolgende Lese- oder Schreibvorgänge aus einem Job mit vollständigem Tabellenzugriff fehl. Das liegt daran, dass AWS Glue Spark ohne vollständigen Tabellenzugriff dem Namen des Tabellenordners das Suffix $folder $ hinzufügt. Um dieses Problem zu lösen, können Sie entweder:
  - Fügen Sie mindestens eine Zeile aus einem Job in die Tabelle ein, für den FTA nicht aktiviert ist.
  - Konfigurieren Sie den Job, für den FTA nicht aktiviert ist, so, dass er das Suffix $folder$ nicht im Ordnernamen in S3 verwendet. Dies kann durch die Einstellung der Spark-Konfiguration erreicht werden. `spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true`
  - Erstellen Sie `s3://path/to/table/table_name` mit der Amazon S3 S3-Konsole oder der Amazon S3 S3-CLI am Speicherort der Tabelle einen Amazon S3 S3-Ordner.
- Full Table Access funktioniert ausschließlich mit dem EMR-Dateisystem (EMRFS). Das S3A-Dateisystem ist nicht kompatibel.
- Full Table Access wird für Hive- und Iceberg-Tabellen unterstützt. Support für Hudi- und Delta-Tabellen wurde noch nicht hinzugefügt.

- Jobs, die auf Tabellen mit FGAC-Regeln (AWS Lake Formation Fine-Grained Access Control) oder AWS Glue Datenkatalogsichten verweisen, schlagen fehl. Um eine Tabelle mit FGAC-Regeln oder einer AWS Glue Datenkatalogsicht abzufragen, müssen Sie den FGAC-Modus verwenden. Sie können den FGAC-Modus aktivieren, indem Sie die in der AWS Dokumentation beschriebenen Schritte ausführen: Verwenden AWS Glue von with für eine detaillierte Zugriffskontrolle. AWS Lake Formation
- Der vollständige Tabellenzugriff unterstützt Spark Streaming nicht.
- Kann nicht gleichzeitig mit FGAC verwendet werden.

Verwendung von AWS Glue mit AWS Lake Formation für eine feinkörnige Zugriffskontrolle

Übersicht

Mit AWS Glue Version 5.0 und höher können Sie detaillierte Zugriffskontrollen auf Datenkatalogtabellen anwenden, die von S3 unterstützt werden. AWS Lake Formation Mit dieser Funktion können Sie Zugriffskontrollen auf Tabellen-, Zeilen-, Spalten- und Zellenebene für read Abfragen innerhalb Ihrer AWS Glue for Apache Spark-Jobs konfigurieren. In den folgenden Abschnitten erfahren Sie mehr über Lake Formation und wie Sie es mit AWS Glue verwenden können.

`GlueContext`Eine basierte Zugriffskontrolle auf Tabellenebene mit AWS Lake Formation Berechtigungen, die in Glue 4.0 oder früher unterstützt wurden, wird in Glue 5.0 nicht unterstützt. Verwenden Sie die neue native Fine-Grained Access Control (FGAC) von Spark in Glue 5.0. Beachten Sie die folgenden Details:

- Wenn Sie Fine Grained Access Control (FGAC) für die row/column/cell Zugriffskontrolle benötigen, müssen Sie in Glue 4.0 von `GlueContext` /Glue und DynamicFrame in Glue 5.0 vor Spark-Dataframe migrieren. Beispiele finden Sie unter Migration von GlueContext/Glue DynamicFrame zu Spark DataFrame
- Wenn Sie Full Table Access Control (FTA) benötigen, können Sie FTA DynamicFrames in Glue 5.0 nutzen. Sie können auch zum nativen Spark-Ansatz migrieren, um zusätzliche Funktionen wie Resilient Distributed Datasets (RDDs), benutzerdefinierte Bibliotheken und benutzerdefinierte Funktionen (UDFs) mit Tabellen zu nutzen. AWS Lake Formation Beispiele finden Sie unter Migration von AWS Glue 4.0 zu AWS Glue 5.0 FTA.

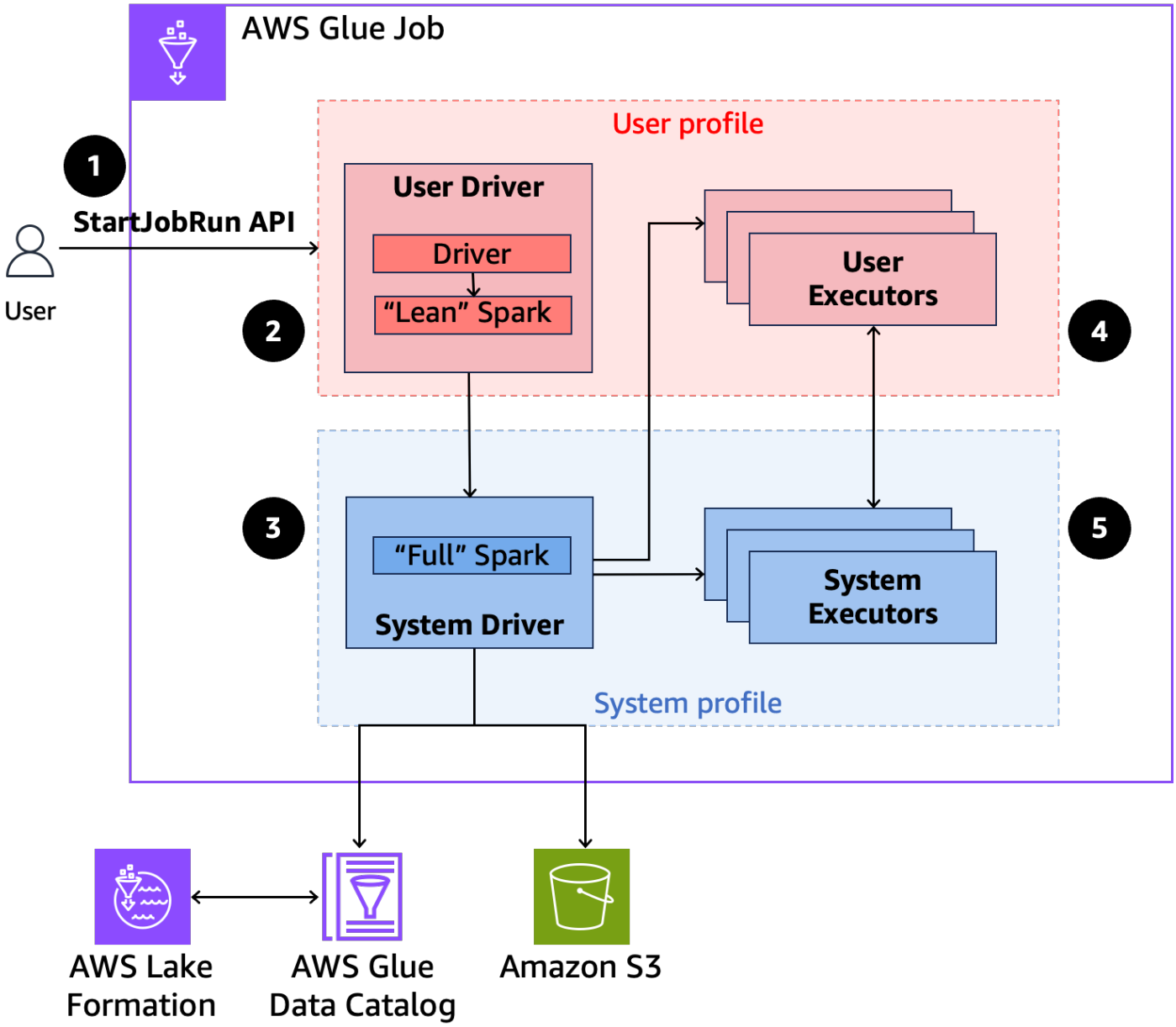
- Wenn Sie FGAC nicht benötigen, ist keine Migration zu Spark-Dataframe erforderlich und `GlueContext` Funktionen wie Job-Lesezeichen und Push-Down-Prädikate funktionieren weiterhin.
- Jobs mit FGAC erfordern mindestens 4 Mitarbeiter: einen Benutzertreiber, einen Systemtreiber, einen System-Executor und einen Standby-User-Executor.

Für die Verwendung von AWS AWS Lake Formation Glue mit fallen zusätzliche Gebühren an.

So funktioniert AWS Glue mit AWS Lake Formation

Wenn Sie AWS Glue mit Lake Formation verwenden, können Sie für jeden Spark-Job eine Berechtigungsebene erzwingen, um die Lake Formation Formation-Berechtigungssteuerung anzuwenden, wenn AWS Glue Jobs ausführt. AWS Glue verwendet Spark-Ressourcenprofile, um zwei Profile für die effektive Ausführung von Jobs zu erstellen. Das Benutzerprofil führt vom Benutzer bereitgestellten Code aus, während das Systemprofil die Lake Formation-Richtlinien durchsetzt. Weitere Informationen finden Sie unter Was ist AWS Lake Formation und Überlegungen und Einschränkungen.

Im Folgenden finden Sie einen allgemeinen Überblick darüber, wie AWS Glue Zugriff auf Daten erhält, die durch die Sicherheitsrichtlinien von Lake Formation geschützt sind.



1. Ein Benutzer ruft die `StartJobRun` API für einen AWS Lake Formation-fähigen AWS Glue-Job auf.
2. AWS Glue sendet den Job an einen Benutzertreiber und führt den Job im Benutzerprofil aus. Der Benutzertreiber führt eine schlanke Version von Spark aus, die nicht in der Lage ist, Aufgaben zu starten, Executors anzufordern, auf S3 oder den Glue-Katalog zuzugreifen. Es erstellt einen Jobplan.
3. AWS Glue richtet einen zweiten Treiber ein, den Systemtreiber, und führt ihn im Systemprofil aus (mit einer privilegierten Identität). AWS Glue richtet einen verschlüsselten TLS-Kanal zwischen den beiden Treibern für die Kommunikation ein. Der Benutzertreiber verwendet den Kanal, um

die Jobpläne an den Systemtreiber zu senden. Der Systemtreiber führt keinen vom Benutzer übermittelten Code aus. Er läuft in vollem Umfang mit Spark und kommuniziert mit S3 und dem Datenkatalog für den Datenzugriff. Es fordert Ausführende an und stellt den Jobplan in eine Abfolge von Ausführungsphasen zusammen.

4. AWS Glue führt dann die Stufen auf Executoren mit dem Benutzertreiber oder Systemtreiber aus. Benutzercode wird in jeder Phase ausschließlich auf Benutzerprofil-Executoren ausgeführt.
5. Stufen, die Daten aus Datenkatalogtabellen lesen, die durch Sicherheitsfilter geschützt sind AWS Lake Formation oder solche, die Sicherheitsfilter anwenden, werden an Systemausführende delegiert.

Mindestanforderung an Arbeitskräfte

Ein Lake Formation-fähiger Job in AWS Glue erfordert mindestens 4 Worker: einen Benutzertreiber, einen Systemtreiber, einen System-Executor und einen Standby-User Executor. Dies ist eine Erhöhung der Mindestanzahl von 2 Arbeitern, die für AWS Standardklebearbeiten erforderlich sind.

Ein Lake Formation-fähiger Job in AWS Glue verwendet zwei Spark-Treiber — einen für das Systemprofil und einen weiteren für das Benutzerprofil. In ähnlicher Weise sind auch die Executors in zwei Profile unterteilt:

- System-Executoren: erledigen Aufgaben, bei denen Lake Formation Formation-Datenfilter angewendet werden.
- Benutzerausführungen: werden bei Bedarf vom Systemtreiber angefordert.

Da Spark-Jobs von Natur aus faul sind, reserviert AWS Glue 10% der gesamten Arbeiter (mindestens 1), nach Abzug der beiden Treiber, für Benutzerausführungen.

Für alle Lake Formation-fähigen Jobs ist die auto-scaling aktiviert, was bedeutet, dass die Benutzerausführungen nur bei Bedarf gestartet werden.

Eine Beispielkonfiguration finden Sie unter Überlegungen und Einschränkungen.

IAM-Berechtigungen für die Rolle „Job Runtime“

Lake Formation Formation-Berechtigungen kontrollieren den Zugriff auf AWS Glue Data Catalog-Ressourcen, Amazon S3 S3-Standorte und die zugrunde liegenden Daten an diesen Standorten. IAM-Berechtigungen kontrollieren den Zugriff auf Lake Formation und AWS Glue APIs sowie auf

Ressourcen. Obwohl Sie möglicherweise über die Lake Formation Formation-Berechtigung verfügen, auf eine Tabelle im Datenkatalog (SELECT) zuzugreifen, schlägt Ihr Vorgang fehl, wenn Sie nicht über die IAM-Berechtigung für den `glue:Get*` API-Vorgang verfügen.

Im Folgenden finden Sie eine Beispielrichtlinie für die Bereitstellung von IAM-Berechtigungen für den Zugriff auf ein Skript in S3, für das Hochladen von Protokollen auf S3, für AWS Glue-API-Berechtigungen und für den Zugriff auf Lake Formation.

JSON

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ScriptAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
            "arn:aws:s3:::*.amzn-s3-demo-bucket/scripts",
            "arn:aws:s3:::*.amzn-s3-demo-bucket/*" ]
        },
        {
         "Sid": "LoggingAccess",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
               "arn:aws:s3:::amzn-s3-demo-bucket/logs/*"
            ]
        },
        {
            "Sid": "GlueCatalogAccess",
            "Effect": "Allow",
            "Action": [
                 "glue:Get*",
                 "glue:Create*",
                 "glue:Update*"
            ],
```

```
            "Resource": ["*"]
        },
        {
            "Sid": "LakeFormationAccess",
            "Effect": "Allow",
            "Action": [
             "lakeformation:GetDataAccess"
             ],
            "Resource": ["*"]
        }
    ]
}
```

Lake Formation Formation-Berechtigungen für die Job-Runtime-Rolle einrichten

Registrieren Sie zunächst den Standort Ihres Hive-Tisches bei Lake Formation. Erstellen Sie dann Berechtigungen für Ihre Job-Runtime-Rolle in der gewünschten Tabelle. Weitere Informationen zu Lake Formation finden Sie unter Was ist AWS Lake Formation? im AWS Lake Formation Entwicklerhandbuch.

Nachdem Sie die Lake Formation Formation-Berechtigungen eingerichtet haben, können Sie Spark-Jobs auf AWS Glue einreichen.

Das Einreichen eines Auftrags wurde ausgeführt

Nachdem Sie die Lake Formation Grants eingerichtet haben, können Sie Spark-Jobs auf AWS Glue einreichen. Um Iceberg-Jobs ausführen zu können, müssen Sie die folgenden Spark-Konfigurationen angeben. Geben Sie zur Konfiguration über Glue-Jobparameter den folgenden Parameter ein:

- Schlüssel:

```
--conf
```

- Wert:

```
spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
       --conf spark.sql.catalog.spark_catalog.warehouse=<S3_DATA_LOCATION>
       --conf spark.sql.catalog.spark_catalog.glue.account-id=<ACCOUNT_ID>
       --conf spark.sql.catalog.spark_catalog.client.region=<REGION>
       --conf spark.sql.catalog.spark_catalog.glue.endpoint=https://
glue.<REGION>.amazonaws.com
```

Verwenden einer interaktiven Sitzung

Nachdem Sie die AWS Lake Formation Grants eingerichtet haben, können Sie interaktive Sessions auf AWS Glue verwenden. Sie müssen die folgenden Spark-Konfigurationen über `%%configure` Magic bereitstellen, bevor Sie den Code ausführen.

```
%%configure
{
    "--enable-lakeformation-fine-grained-access": "true",
    "--conf":
 "spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
 --conf spark.sql.catalog.spark_catalog.warehouse=<S3_DATA_LOCATION> --conf
 spark.sql.catalog.spark_catalog.catalog-impl=org.apache.iceberg.aws.glue.GlueCatalog
 --conf spark.sql.catalog.spark_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO --
conf
 spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 --conf spark.sql.catalog.spark_catalog.client.region=<REGION> --conf
 spark.sql.catalog.spark_catalog.glue.account-id=<ACCOUNT_ID> --conf
 spark.sql.catalog.spark_catalog.glue.endpoint=https://glue.<REGION>.amazonaws.com"
}
```

FGAC für AWS Glue 5.0 Notebook oder interaktive Sitzungen

Um Fine-Grained Access Control (FGAC) in AWS Glue zu aktivieren, müssen Sie die für Lake Formation erforderlichen Spark-Konfigurationen als Teil von %%configure Magic angeben, bevor Sie die erste Zelle erstellen.

Es reicht nicht aus, es später mit den Aufrufen `SparkSession.builder().conf("").get()` oder anzugeben. `SparkSession.builder().conf("").create()` Dies ist eine Änderung gegenüber dem Verhalten von AWS Glue 4.0.

Unterstützung für das Open-Table-Format

AWS Glue Version 5.0 oder höher bietet Unterstützung für eine feinkörnige Zugriffskontrolle auf Basis von Lake Formation. AWS Glue unterstützt die Tabellentypen Hive und Iceberg. In der folgenden Tabelle werden alle unterstützten Operationen beschrieben.

Operationen	Hive	Iceberg
DDL-Befehle	Nur mit IAM-Rollenberechtigungen	Nur mit IAM-Rollenberechtigungen

Operationen	Hive	Iceberg
Inkrementelle Abfragen	Nicht zutreffend	Vollständig unterstützt
Zeitreiseabfragen	Gilt nicht für dieses Tabellenformat	Vollständig unterstützt
Metadaten-Tabellen	Gilt nicht für dieses Tabellenformat	Wird unterstützt, aber bestimmte Tabellen sind ausgeblendet. Weitere Informationen finden Sie unter Überlegungen und Einschränkungen.
`DML INSERT`	Nur mit IAM-Berechtigungen	Nur mit IAM-Berechtigungen
DML-UPDATE	Gilt nicht für dieses Tabellenformat	Nur mit IAM-Berechtigungen
`DML DELETE`	Gilt nicht für dieses Tabellenformat	Nur mit IAM-Berechtigungen
Lesevorgänge	Vollständig unterstützt	Vollständig unterstützt
Gespeicherte Prozeduren	Nicht zutreffend	Unterstützt mit den Ausnahmen von `register_table` und `migrate`. Weitere Informationen finden Sie unter Überlegungen und Einschränkungen.

Migration von GlueContext/Glue DynamicFrame zu Spark DataFrame

Im Folgenden finden Sie Python- und Scala-Beispiele für die Migration von `GlueContext` /Glue `DynamicFrame` in Glue 4.0 zu Spark `DataFrame` in Glue 5.0.

Python

Vorher:

```
escaped_table_name= '`<dbname>`.`<table_name>`'

additional_options = {
  "query": f'select * from {escaped_table_name} WHERE column1 = 1 AND column7 = 7'
}

# DynamicFrame example
dataset = glueContext.create_data_frame_from_catalog(
    database="<dbname>",
    table_name=escaped_table_name,
    additional_options=additional_options)
```

Nachher:

```
table_identifier= '`<catalogname>`.`<dbname>`.`<table_name>`"' #catalogname is optional

# DataFrame example
dataset = spark.sql(f'select * from {table_identifier} WHERE column1 = 1 AND column7 =
 7')
```

Scala

Vorher:

```
val escapedTableName = "`<dbname>`.`<table_name>`"

val additionalOptions = JsonOptions(Map(
    "query" -> s"select * from $escapedTableName WHERE column1 = 1 AND column7 = 7"
    )
)

# DynamicFrame example
val datasource0 = glueContext.getCatalogSource(
    database="<dbname>",
    tableName=escapedTableName,
    additionalOptions=additionalOptions).getDataFrame()
```

Nachher:

```
val tableIdentifier = "`<catalogname>`.`<dbname>`.`<table_name>`" //catalogname is
 optional
```

```
# DataFrame example
val datasource0 = spark.sql(s"select * from $tableIdentifier WHERE column1 = 1 AND
 column7 = 7")
```

Überlegungen und Einschränkungen

Beachten Sie die folgenden Überlegungen und Einschränkungen, wenn Sie Lake Formation mit AWS Glue verwenden.

AWS Glue with Lake Formation ist in allen unterstützten Regionen außer AWS GovCloud (US-Ost) und AWS GovCloud (US-West) verfügbar.

- AWS Glue unterstützt eine feinkörnige Zugriffskontrolle über Lake Formation nur für Apache Hive- und Apache Iceberg-Tabellen. Zu den Apache Hive-Formaten gehören Parquet, ORC und CSV.
- Sie können Lake Formation nur mit Spark-Jobs verwenden.
- AWS Glue mit Lake Formation unterstützt nur eine einzige Spark-Sitzung während eines Jobs.
- Wenn Lake Formation aktiviert ist, benötigt AWS Glue eine größere Anzahl von Workern, da es einen Systemtreiber, Systemausführungen, einen Benutzertreiber und optional Benutzerausführungen benötigt (erforderlich, wenn Ihr Job oder hat). UDFs `spark.createDataFrame`
- AWS Glue with Lake Formation unterstützt nur kontenübergreifende Tabellenabfragen, die über Ressourcenlinks gemeinsam genutzt werden. Der Ressourcenlink muss identisch mit der Ressource des Quellkontos benannt werden.
- Um eine differenzierte Zugriffskontrolle für AWS Glue-Jobs zu aktivieren, übergeben Sie den `--enable-lakeformation-fine-grained-access` Job-Parameter.
- Sie können Ihre AWS Glue-Jobs so konfigurieren, dass sie mit der AWS Glue-Hierarchie mit mehreren Katalogen funktionieren. Informationen zu den Konfigurationsparametern, die mit der AWS `StartJobRun` Glue-API verwendet werden können, finden Sie unter Arbeiten mit der AWS Glue-Multi-Katalog-Hierarchie auf EMR Serverless.
- Folgendes wird nicht unterstützt:
  - Resilient Distributed Datasets (RDD)
  - Spark-Streaming
  - Schreiben Sie mit Lake Formation erteilten Berechtigungen
  - Zugriffskontrolle für verschachtelte Spalten

- AWS Glue blockiert Funktionen, die die vollständige Isolierung des Systemtreibers untergraben könnten, darunter die folgenden:
  - UDTs, Hive und alle benutzerdefinierten FunktionenUDFs, die benutzerdefinierte Klassen beinhalten
  - Benutzerdefinierte Datenquellen
  - Bereitstellung zusätzlicher Jars für Spark-Erweiterungen, Konnektoren oder Metastore
  - `ANALYZE TABLE` command
- Um Zugriffskontrollen `EXPLAIN PLAN` und DDL-Operationen durchzusetzen, z. B. um eingeschränkte `DESCRIBE TABLE` Informationen nicht preiszugeben.
- AWS Glue schränkt den Zugriff auf Systemtreiber-Spark-Logs für Lake Formation-fähige Anwendungen ein. Da der Systemtreiber mit mehr Zugriff ausgeführt wird, können Ereignisse und Protokolle, die der Systemtreiber generiert, vertrauliche Informationen enthalten. Um zu verhindern, dass unbefugte Benutzer oder Code auf diese sensiblen Daten zugreifen, hat AWS Glue den Zugriff auf Systemtreiberprotokolle deaktiviert. Wenden Sie sich zur Fehlerbehebung an den AWS Support.
- Wenn Sie einen Tabellenstandort bei Lake Formation registriert haben, durchläuft der Datenzugriffspfad unabhängig von der IAM-Berechtigung für die AWS Glue-Job-Laufzeitrolle die gespeicherten Anmeldeinformationen von Lake Formation. Wenn Sie die mit der Tabellenposition registrierte Rolle falsch konfigurieren, schlagen gesendete Jobs fehl, die die Rolle mit der S3-IAM-Berechtigung für den Tabellenspeicherort verwenden.
- Beim Schreiben in eine Lake Formation-Tabelle werden IAM-Berechtigungen und nicht die von Lake Formation erteilten Berechtigungen verwendet. Wenn Ihre Job-Runtime-Rolle über die erforderlichen S3-Berechtigungen verfügt, können Sie sie zum Ausführen von Schreibvorgängen verwenden.

Im Folgenden finden Sie Überlegungen und Einschränkungen bei der Verwendung von Apache Iceberg:

- Sie können Apache Iceberg nur mit Sitzungskatalogen und nicht mit beliebig benannten Katalogen verwenden.
- Iceberg-Tabellen, die in Lake Formation registriert sind, unterstützen nur die Metadatentabellen `history`,`metadata_log_entries`,`snapshots`,`files`,`manifests`, und`refs`. AWS Glue blendet die Spalten aus, die möglicherweise vertrauliche Daten enthalten, wie `partitions`, und`summaries`. Diese Einschränkung gilt nicht für Iceberg-Tabellen, die nicht in Lake Formation registriert sind.

- Tabellen, die Sie nicht in Lake Formation registrieren, unterstützen alle gespeicherten Iceberg-Prozeduren. Die `migrate` Prozeduren `register_table` und werden für keine Tabellen unterstützt.
- Wir empfehlen, Iceberg DataFrameWriter V2 anstelle von V1 zu verwenden.

Beispiel für eine Mitarbeiterzuweisung

Für einen Job, der mit den folgenden Parametern konfiguriert wurde:

```
--enable-lakeformation-fine-grained-access=true
--number-of-workers=20
```

Die Zuteilung der Arbeitskräfte würde wie folgt aussehen:

- Ein Arbeiter für den Benutzerfahrer.
- Ein Arbeiter für den Systemtreiber.
- 10% der verbleibenden 18 Mitarbeiter (d. h. 2 Mitarbeiter) sind für die Ausführung durch den Benutzer reserviert.
- Bis zu 16 Mitarbeiter sind für System-Executoren vorgesehen.

Wenn die auto-scaling aktiviert ist, können die Benutzerausführungen bei Bedarf jede der nicht zugewiesenen Kapazitäten der System-Executoren nutzen.

Steuerung der Zuweisung von Benutzer-Executoren

Sie können den Reservierungsprozentsatz für Benutzer-Executors mithilfe der folgenden Konfiguration anpassen:

```
--conf spark.dynamicAllocation.maxExecutorsRatio=<value between 0 and 1>
```

Diese Konfiguration ermöglicht eine genaue Steuerung darüber, wie viele Benutzer-Executors im Verhältnis zur verfügbaren Gesamtkapazität reserviert werden.

Fehlerbehebung

In den folgenden Abschnitten finden Sie Lösungen zur Fehlerbehebung.

Protokollierung

AWS Glue verwendet Spark-Ressourcenprofile, um die Auftragsausführung aufzuteilen. AWS Glue verwendet das Benutzerprofil, um den von Ihnen bereitgestellten Code auszuführen, während das Systemprofil die Lake Formation-Richtlinien durchsetzt. Sie können auf die Protokolle für die Aufgaben zugreifen, die als Benutzerprofil ausgeführt wurden.

Live-Benutzeroberfläche und Spark History Server

In der Live-Benutzeroberfläche und auf dem Spark History Server werden alle Spark-Ereignisse aus dem Benutzerprofil und aus dem Systemtreiber redigierte Ereignisse generiert.

Sie können alle Aufgaben sowohl der Benutzer- als auch der Systemtreiber auf der Registerkarte Executors sehen. Protokolllinks sind jedoch nur für das Benutzerprofil verfügbar. Außerdem werden einige Informationen aus der Live-Benutzeroberfläche entfernt, z. B. die Anzahl der Ausgabedatensätze.

Job mit unzureichenden Lake Formation Formation-Berechtigungen fehlgeschlagen

Stellen Sie sicher, dass Ihre Job-Runtime-Rolle über die Berechtigungen zum Ausführen von SELECT und DESCRIBE für die Tabelle verfügt, auf die Sie zugreifen.

Job mit RDD-Ausführung ist fehlgeschlagen

AWS Glue unterstützt derzeit keine Resilient Distributed Dataset (RDD) -Operationen für Lake Formation-fähige Jobs.

Auf Datendateien in Amazon S3 kann nicht zugegriffen werden

Stellen Sie sicher, dass Sie den Standort des Data Lake in Lake Formation registriert haben.

Ausnahme bei der Sicherheitsüberprüfung

AWS Glue hat einen Fehler bei der Sicherheitsüberprüfung festgestellt. Wenden Sie sich an den AWS Support, um Unterstützung zu erhalten

Gemeinsame Nutzung des AWS Glue-Datenkatalogs und der Tabellen für mehrere Konten

Sie können Datenbanken und Tabellen für mehrere Konten gemeinsam nutzen und trotzdem Lake Formation verwenden. Weitere Informationen finden Sie unter Kontoübergreifender Datenaustausch

in Lake Formation und Wie teile ich den AWS Glue-Datenkatalog und die Tabellen kontoübergreifend mit? .

In der folgenden Tabelle wird zusammenfassend beschrieben, wie Sie für Ihren Workload zwischen Fine-Grained Access Control (FGAC) und Full Table Access (FTA) wählen können.

Funktion	Differenzierte Zugriffskontrolle (FGAC, Fine-Grained Access Control)	Vollständiger Tabellenzugriff (FTA)
Zugriffsebene	Spalten-/Zeilenebene	Vollständige Tabelle
Anwendungsfall	Abfragen und ETL mit eingeschränkten Berechtigungen	ETL
Auswirkung auf die Leistung	Erfordert system/user Speicherübergänge für die Evaluierung der Zugriffskontrolle, was die Latenz erhöht	Optimierte Leistung

Verwenden von Amazon S3 Access Grants mit AWS Glue

Mit Glue Version 5.0 bieten Amazon S3 Access Grants eine skalierbare Zugriffskontrolllösung, mit der Sie den Zugriff auf Ihre Amazon S3 S3-Daten erweitern können. AWS Glue Wenn Sie über eine komplexe oder umfangreiche Berechtigungskonfiguration für Ihre S3-Daten verfügen, können Sie S3 Access Grants verwenden, um S3-Datenberechtigungen für Benutzer und Rollen zu skalieren.

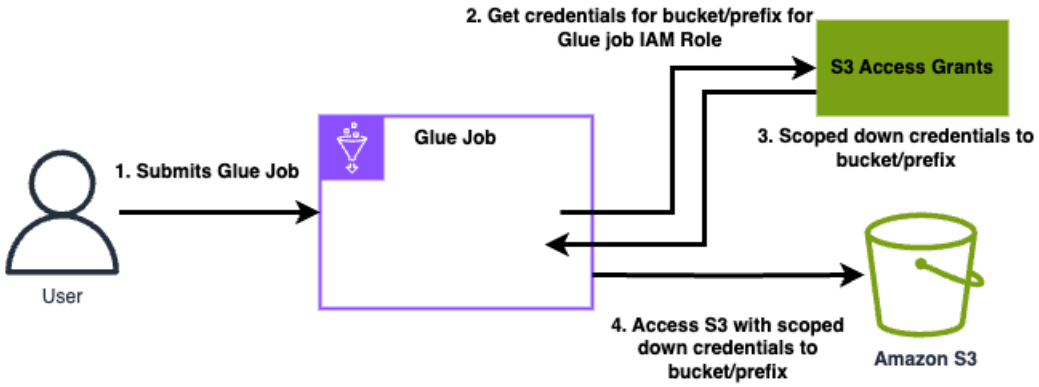
Verwenden Sie S3 Access Grants, um den Zugriff auf Amazon S3 S3-Daten über die Berechtigungen hinaus zu erweitern, die durch die Runtime-Rolle oder die IAM-Rollen gewährt werden, die den Identitäten mit Zugriff auf Ihren Job zugewiesen sind. AWS Glue Weitere Informationen finden Sie unter Verwalten des Zugriffs mit S3 Access Grants im Benutzerhandbuch zu Amazon S3.

Wie AWS Glue funktioniert mit S3 Access Grants

AWS Glue Versionen 5.0 und höher bieten eine native Integration mit S3 Access Grants. Sie können S3 Access Grants aktivieren AWS Glue und Spark-Jobs ausführen. Wenn ein Spark-Auftrag eine

Anfrage für S3-Daten stellt, stellt Amazon S3 temporäre Anmeldeinformationen bereit, die auf den jeweiligen Bucket, das Präfix oder das Objekt beschränkt sind.

Im Folgenden finden Sie einen allgemeinen Überblick darüber, wie Sie AWS Glue auf Daten zugreifen können, auf die S3 Access Grants den Zugriff verwaltet.



1. Ein Benutzer reicht einen AWS Glue Spark-Job ein, der in Amazon S3 gespeicherte Daten verwendet.
2. AWS Glue stellt eine Anfrage für S3 Access Grants, um temporäre Anmeldeinformationen für den Benutzer zu verkaufen, die Zugriff auf den Bucket, das Präfix oder das Objekt gewähren.
3. AWS Glue gibt temporäre Anmeldeinformationen in Form eines STS-Tokens (AWS Security Token Service) für den Benutzer zurück. Das Token ist für den Zugriff auf den S3-Bucket, das S3-Präfix oder das S3-Objekt vorgesehen.
4. AWS Glue verwendet das STS-Token, um Daten aus S3 abzurufen.
5. AWS Glue empfängt die Daten von S3 und gibt die Ergebnisse an den Benutzer zurück.

Überlegungen zu S3 Access Grants mit AWS Glue

Beachten Sie die folgenden Verhaltensweisen und Einschränkungen, wenn Sie S3 Access Grants mit verwenden AWS Glue.

Unterstützung von Funktionen

- S3 Access Grants wird mit AWS Glue Versionen 5.0 und höher unterstützt.
- Spark ist der einzige unterstützte Jobtyp, wenn Sie S3 Access Grants mit verwenden AWS Glue.
- Delta Lake und Hudi sind die einzigen unterstützten Open-Table-Formate, wenn Sie S3 Access Grants mit verwenden. AWS Glue

- Die folgenden Funktionen werden für die Verwendung mit S3 Access Grants nicht unterstützt:
  - Apache-Iceberg-Tabellen
  - AWS CLI-Anfragen an Amazon S3, die IAM-Rollen verwenden
  - S3-Zugriff über das Open-Source-S3A-Protokoll

Überlegungen zum Verhalten

- AWS Glue stellt einen Cache für Anmeldeinformationen bereit, um sicherzustellen, dass ein Benutzer innerhalb eines Spark-Jobs nicht wiederholt dieselben Anmeldeinformationen anfordern muss. Fordert daher AWS Glue immer das Standardrecht an, wenn es Anmeldeinformationen anfordert. Weitere Informationen finden Sie unter Zugriff auf S3-Daten anfordern im Benutzerhandbuch zu Amazon S3.

Richten Sie S3 Access Grants ein mit AWS Glue

Voraussetzungen

Der Anrufer oder Administrator hat eine S3 Access Grants-Instanz erstellt.

Richten Sie AWS Glue Richtlinien und Auftragskonfigurationen ein

Um S3 Access Grants mit einzurichten, müssen AWS Glue Sie Vertrauens- und IAM-Richtlinien konfigurieren und die Konfiguration über die Jobparameter übergeben.

1. Konfigurieren Sie die folgenden minimalen Vertrauens- und IAM-Richtlinien für die Rolle, die für Grants verwendet wird (die AWS Glue Rolle, die Sitzungen oder Jobs ausführt).

   Vertrauensrichtlinie:

```
{
            "Sid": "Stmt1234567891011",
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity",
                "sts:SetContext"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": "access-grants.s3.amazonaws.com"
            },
```

```
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:s3:<region>:123456789012:access-grants/
default"
                }
            }
        }
```

IAM-Richtlinie:

```
{
            "Sid": "S3Grants",
            "Effect": "Allow",
            "Action": [
                "s3:GetDataAccess",
                "s3:GetAccessGrantsInstanceForPrefix"
            ],
            "Resource": "arn:aws:s3:<region>:123456789012:access-grants/default"
        },
        {
            "Sid": "BucketLevelReadPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                },
                "ArnEquals": {
                    "s3:AccessGrantsInstanceArn": [
                        "arn:aws:s3:<region>:123456789012:access-grants/default"
                    ]
                }
            }
        },
        {
            "Sid": "ObjectLevelReadPermissions",
            "Effect": "Allow",
```

```
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetObjectAcl",
                "s3:GetObjectVersionAcl",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                },
                "ArnEquals": {
                    "s3:AccessGrantsInstanceArn": [
                        "arn:aws:s3:<region>:123456789012:access-grants/default"
                    ]
                }
            }
        }
```

2. Übergeben Sie in Ihrem AWS Glue Job die folgende Spark-Konfiguration entweder über die AWS Glue Job-Parameter oder`SparkConf`.

```
--conf spark.hadoop.fs.s3.s3AccessGrants.enabled=true \
--conf spark.hadoop.fs.s3.s3AccessGrants.fallbackToIAM=false
```

Einloggen und Überwachen AWS Glue

Sie können die Ausführung Ihrer ETL-Jobs (Extrahieren, Transformieren und Laden) automatisieren. AWS Glue bietet Metriken für Crawler und Jobs, die Sie überwachen können. Nachdem Sie das AWS Glue Data Catalog mit den erforderlichen Metadaten eingerichtet haben, AWS Glue stellt Statistiken über den Zustand Ihrer Umgebung bereit. Sie können das Aufrufen der Crawler und Aufträge mit einem zeitbasierten Plan auf Basis von Cron automatisieren. Außerdem können Sie Aufträge auslösen, wenn ein ereignisbasierter Auslöser aktiviert wird.

AWS Glue ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in bereitstellt AWS Glue. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon

Simple Storage Service (Amazon S3) -Bucket, Amazon CloudWatch Logs und Amazon CloudWatch Events aktivieren. Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat.

Verwenden Sie Amazon CloudWatch Events, um Ihre AWS Services zu automatisieren und automatisch auf Systemereignisse wie Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen zu reagieren. Ereignisse aus AWS Services werden nahezu in Echtzeit an CloudWatch Events übermittelt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen durchgeführt werden sollen, wenn sich für ein Ereignis eine Übereinstimmung mit einer Regel ergibt.

Weitere Informationen finden Sie auch unter

- Automatisieren AWS Glue mit EventBridge
- Kontoübergreifende Protokollierung CloudTrail

Ein wichtiger Aspekt der Sicherheit in der Cloud ist die Protokollierung. Sie müssen die Protokollierung so konfigurieren, dass keine geheimen und vertraulichen Daten erfasst werden, während Informationen erfasst werden, die zur Fehlersuche und Sicherung Ihrer Cloud-Infrastruktur erforderlich sind. Machen Sie sich mit dem, was protokolliert wird, vertraut.

Konformitätsvalidierung für AWS Glue

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- Compliance und Governance im Bereich Sicherheit – In diesen Anleitungen für die Lösungsimplementierung werden Überlegungen zur Architektur behandelt. Außerdem werden Schritte für die Bereitstellung von Sicherheits- und Compliance-Features beschrieben.
- Referenz für berechtigte HIPAA-Services – Listet berechtigte HIPAA-Services auf. Nicht alle AWS-Services sind HIPAA-fähig.
- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- AWS Leitfäden zur Einhaltung von Vorschriften für Kunden — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- Evaluierung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub— Dies AWS-Service bietet einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Die Liste der unterstützten Services und Kontrollen finden Sie in der Security-Hub-Steuerelementreferenz.
- Amazon GuardDuty — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- AWS Audit Manager— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

Resilienz in AWS Glue

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. AWS Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability

Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Availability Zones ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser hoch verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Weitere Informationen zur Ausfallsicherheit von AWS Glue Jobs finden Sie unter Fehler: Failover-Verhalten zwischen VPCs . AWS Glue

Infrastruktursicherheit in AWS Glue

Als verwalteter Dienst AWS Glue ist durch die AWS globalen Netzwerksicherheitsverfahren geschützt, die im Whitepaper Amazon Web Services: Sicherheitsprozesse im Überblick beschrieben sind.

Sie verwenden AWS veröffentlichte API-Aufrufe für den Zugriff AWS Glue über das Netzwerk. Kunden müssen Transport Layer Security (TLS) 1.0 oder neuer unterstützen. Wir empfehlen TLS 1.2 oder höher. Clients müssen außerdem Verschlüsselungssammlungen mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Themen

- Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS PrivateLink für AWS Glue (AWS PrivateLink)
- Konfiguration von Shared Amazon VPCs

Konfiguration von Schnittstellen-VPC-Endpunkten ()AWS PrivateLink für AWS Glue (AWS PrivateLink)

Sie können eine private Verbindung zwischen Ihrer VPC herstellen und AWS Glue indem Sie einen VPC-Schnittstellen-Endpunkt erstellen. Schnittstellen-Endpunkte werden mit einer Technologie

betrieben AWS PrivateLink, mit der Sie privat darauf zugreifen können AWS Glue APIs ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder AWS Direct Connect-Verbindung. Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen für die Kommunikation mit AWS Glue APIs. Verkehr zwischen Ihrer VPC und AWS Glue verlässt das Amazon-Netzwerk nicht.

Jeder Schnittstellenendpunkt wird durch eine oder mehrere Elastic-Network-Schnittstellen in Ihren Subnetzen dargestellt.

Weitere Informationen finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon-VPC-Benutzerhandbuch.

Überlegungen zu AWS Glue VPC-Endpunkte

Bevor Sie einen VPC-Schnittstellen-Endpunkt für einrichten AWS Glue, stellen Sie sicher, dass Sie die Eigenschaften und Einschränkungen der Schnittstellenendpunkte im Amazon VPC-Benutzerhandbuch lesen.

AWS Glue unterstützt Aufrufe aller API-Aktionen von Ihrer VPC aus.

Erstellen eines Schnittstellen-VPC-Endpunkts für AWS Glue

Sie können einen VPC-Endpunkt erstellen für AWS Glue Service, der entweder die Amazon VPC-Konsole oder die AWS Command Line Interface (AWS CLI) verwendet. Weitere Informationen finden Sie unter Erstellung eines Schnittstellenendpunkts im Benutzerhandbuch für Amazon VPC.

Erstellen Sie einen VPC-Endpunkt für AWS Glue unter Verwendung des folgenden Dienstnamens:

- com.amazonaws. *region*. kleben

Wenn Sie privates DNS für den Endpunkt aktivieren, können Sie API-Anfragen stellen an AWS Glue unter Verwendung des Standard-DNS-Namens für die Region, `glue.us-east-1.amazonaws.com` z. B.

Weitere Informationen finden Sie unter Zugriff auf einen Service über einen Schnittstellenendpunkt im Benutzerhandbuch für Amazon VPC.

Erstellen einer VPC-Endpunktrichtlinie für AWS Glue

Sie können Ihrem VPC-Endpunkt eine Endpunktrichtlinie hinzufügen, die den Zugriff auf AWS Glue. Die Richtlinie spezifiziert die folgenden Informationen:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Die Ressourcen, für die Aktionen ausgeführt werden können.

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Beispiel: VPC-Endpunktrichtlinie für AWS Glue um die Schaffung und Aktualisierung von Arbeitsplätzen zu ermöglichen

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für AWS Glue. Wenn diese Richtlinie an einen Endpunkt angehängt ist, gewährt sie Zugriff auf die aufgelisteten AWS Glue Aktionen für alle Principals auf allen Ressourcen.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
             "glue:CreateJob",
             "glue:UpdateJob",
             "iam:PassRole"
         ],
         "Resource":"*"
      }
   ]
}
```

Beispiel: VPC-Endpunktrichtlinie, um schreibgeschützten Data Catalog-Zugriff zu erlauben

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für AWS Glue. Wenn diese Richtlinie an einen Endpunkt angehängt ist, gewährt sie Zugriff auf die aufgelisteten AWS Glue Aktionen für alle Principals auf allen Ressourcen.

```
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
```

```
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition",
                "glue:SearchTables"
            ],
            "Resource": "*"
        }
    ]
}
```

Konfiguration von Shared Amazon VPCs

AWS Glue unterstützt gemeinsam genutzte virtuelle private Clouds (VPCs) in Amazon Virtual Private Cloud. Amazon VPC Sharing ermöglicht es mehreren AWS Konten, ihre Anwendungsressourcen, wie EC2 Amazon-Instances und Amazon Relational Database Service (Amazon RDS) -Datenbanken, in gemeinsam genutztem, zentral verwaltetem Amazon zu erstellen. VPCs In diesem Modell teilt sich das Konto, dem die VPC gehört (Eigentümer), ein oder mehrere Subnetze mit anderen Konten (Teilnehmern), die derselben Organisation angehören. AWS Organizations Wenn ein Subnetz freigegeben wurde, können die Teilnehmer ihre Anwendungsressourcen in den für sie freigegebenen Subnetzen anzeigen, erstellen, ändern und löschen.

Um eine Verbindung mit einem gemeinsam genutzten Subnetz herzustellen, müssen Sie in Ihrem Konto eine Sicherheitsgruppe erstellen und die Sicherheitsgruppe an das gemeinsam genutzte Subnetz anhängen. AWS Glue

Weitere Informationen enthalten die folgenden Themen:

- Arbeiten mit Shared VPCs im Amazon VPC-Benutzerhandbuch
- Was ist AWS Organizations? im AWS Organizations Benutzerhandbuch

Fehlerbehebung AWS Glue

Wenn Sie bei der Arbeit mit auf Probleme stoßen AWS Glue, lesen Sie die Themen in diesem Abschnitt.

Themen

- Informationen AWS Glue zur Problembehandlung sammeln
- Fehler beheben Spark-Fehler
- Crawler-Fehler, wenn der Crawler Lake Formation Formation-Berechtigungen verwendet
- Fehlerbehebung AWS Glue bei Ray-Fehlern aus Protokollen
- AWS Glue Ausnahmen für maschinelles Lernen
- AWS Glue Kontingente

Informationen AWS Glue zur Problembehandlung sammeln

Wenn Sie auf Fehler oder unerwartetes Verhalten stoßen in AWS Glue und wenn Sie Kontakt aufnehmen müssen AWS -Support, sollten Sie zunächst Informationen zu Namen und Protokollen sammeln IDs, die mit der fehlgeschlagenen Aktion verknüpft sind. Wenn Sie diese Informationen zur Verfügung haben Support , können wir Ihnen helfen, die aufgetretenen Probleme zu lösen.

Sammeln Sie zusammen mit Ihrer Konto-ID die folgenden Informationen für die Fehlerarten:

Wenn ein Crawler fehlschlägt, sammeln Sie die folgenden Informationen:

- Crawler-Name

  Protokolle von Crawler-Läufen befinden sich in CloudWatch Protokollen unter`/aws-glue/crawlers`.

Wenn eine Testverbindung fehlschlägt, sammeln Sie die folgenden Informationen:

- Verbindungsname
- Verbindungs-ID
- JDBC-Verbindungsstring in der Form `jdbc:protocol://host:port/database-name`.

  Protokolle von Testverbindungen befinden sich in den CloudWatch Protokollen unter `/aws-glue/testconnection`.

Wenn ein Auftrag fehlschlägt, sammeln Sie die folgenden Informationen:

- Job name (Auftragsname)
- Auftragsausführungs-ID in der Form `jr_xxxxx`.

  Protokolle von Auftragsausführungen befinden sich in CloudWatch Protokollen unter `/aws-glue/jobs`.

Fehler beheben Spark-Fehler

Wenn Sie in auf Fehler stoßen AWS Glue, verwenden Sie die folgenden Informationen, um die Ursache der Probleme zu finden und sie zu beheben.

> **Note**
>
> Das AWS Glue GitHub Repository enthält unter AWS Glue Häufig gestellte Fragen zusätzliche Anleitungen zur Fehlerbehebung.

Themen

- Fehler: Ressource nicht verfügbar
- Fehler: S3-Endpunkt oder NAT-Gateway für SubnetId wurde in VPC nicht gefunden
- Fehler: Eingehende Regel in der Sicherheitsgruppe erforderlich
- Fehler: Ausgehende Regel in der Sicherheitsgruppe erforderlich
- Fehler: Die Auftragsausführung ist fehlgeschlagen, da der übergebenen Rolle die Rollenübernahme für den AWS Glue Dienst erteilt werden sollte
- Fehler: Die DescribeVpcEndpoints Aktion ist nicht autorisiert. die VPC-ID kann nicht validiert werden vpc-id
- Fehler: Die DescribeRouteTables Aktion ist nicht autorisiert. Die Subnetz-ID konnte nicht überprüft werden: Subnetz-ID in VPC-ID: vpc-id
- Fehler: ec2 konnte nicht aufgerufen werden: DescribeSubnets
- Fehler: ec2 konnte nicht aufgerufen werden: DescribeSecurityGroups
- Fehler: Subnetz für AZ wurde nicht gefunden
- Fehler: Ausnahme der Auftragsausführung beim Schreiben in ein JDBC-Ziel

- Fehler: Amazon S3: Der Vorgang ist für die Speicherklasse des Objekts nicht gültig
- Fehler: Amazon S3 Zeitüberschreitung
- Fehler: Amazon-S3-Zugriff verweigert
- Fehler: Amazon-S3-Zugriffsschlüssel-ID ist nicht vorhanden
- Fehler: Auftragsausführung schlägt fehl, wenn auf Amazon S3 mit einem s3a://-URI zugegriffen wird
- Fehler: Amazon-S3-Servicetoken abgelaufen
- Fehler: Keine private DNS für die Netzwerkschnittstelle gefunden
- Fehler: Bereitstellung des Entwicklungsendpunkts fehlgeschlagen
- Fehler: Notebook-Server-ERSTELLUNG_FEHLGESCHLAGEN
- Fehler: Lokales Notebook kann nicht gestartet werden
- Fehler: Ausführen von Crawler fehlgeschlagen
- Fehler: Partitionen wurden nicht aktualisiert
- Fehler: Aktualisierung des Auftragslesezeichens aufgrund einer Versionsabweichung fehlgeschlagen
- Fehler: Ein Auftrag verarbeitet Daten erneut, wenn Auftragslesezeichen aktiviert sind
- Fehler: Failover-Verhalten zwischen VPCs AWS Glue

Fehler: Ressource nicht verfügbar

Wenn AWS Glue die Meldung „Ressource nicht verfügbar“ angezeigt wird, können Sie sich Fehlermeldungen oder Protokolle ansehen, um mehr über das Problem zu erfahren. Die folgenden Aufgaben beschreiben allgemeine Methoden zur Fehlerbehebung.

- Überprüfen Sie bei allen von Ihnen genutzten Verbindungen und Entwicklungsendpunkten, ob Ihr Cluster über genügend Elastic Network-Schnittstellen verfügt.

Fehler: S3-Endpunkt oder NAT-Gateway für SubnetId wurde in VPC nicht gefunden

Prüfen Sie die Subnetz- und VPC-ID in der Nachricht, die Ihnen dabei helfen, das Problem zu diagnostizieren.

- Stellen Sie sicher, dass Sie einen Amazon-S3-VPC-Endpunkt einrichtet haben, der für AWS Glue erforderlich ist. Überprüfen Sie außerdem Ihr NAT-Gateway, wenn das Teil Ihrer Konfiguration ist. Weitere Informationen finden Sie unter Amazon-VPC-Endpunkte für Amazon S3.

Fehler: Eingehende Regel in der Sicherheitsgruppe erforderlich

Mindestens eine Sicherheitsgruppe muss alle Eingangs-Ports öffnen. Um Datenverkehr zu begrenzen, kann die Quell-Sicherheitsgruppe in Ihrer eingehenden Regel auf dieselbe Sicherheitsgruppe beschränkt werden.

- Prüfen Sie bei allen genutzten Verbindungen Ihre Sicherheitsgruppe auf eine eingehende Regel, die selbst verweisend ist. Weitere Informationen finden Sie unter Netzwerkzugriff auf Datenspeicher einrichten.
- Wenn Sie einen Entwicklungsendpunkt verwenden, überprüfen Sie Ihre Sicherheitsgruppe auf eine eingehende Regel, die selbst verweisend ist. Weitere Informationen finden Sie unter Netzwerkzugriff auf Datenspeicher einrichten.

Fehler: Ausgehende Regel in der Sicherheitsgruppe erforderlich

Mindestens eine Sicherheitsgruppe muss alle Ausgangs-Ports öffnen. Um Datenverkehr zu begrenzen, kann die Quell-Sicherheitsgruppe in Ihrer ausgehenden Regel auf dieselbe Sicherheitsgruppe beschränkt werden.

- Prüfen Sie bei allen genutzten Verbindungen Ihre Sicherheitsgruppe auf eine ausgehende Regel, die selbst verweisend ist. Weitere Informationen finden Sie unter Netzwerkzugriff auf Datenspeicher einrichten.
- Wenn Sie einen Entwicklungsendpunkt verwenden, überprüfen Sie Ihre Sicherheitsgruppe auf eine ausgehende Regel, die selbst verweisend ist. Weitere Informationen finden Sie unter Netzwerkzugriff auf Datenspeicher einrichten.

Fehler: Die Auftragsausführung ist fehlgeschlagen, da der übergebenen Rolle die Rollenübernahme für den AWS Glue Dienst erteilt werden sollte

Der Benutzer, der einen Auftrag definiert, muss über Berechtigungen für `iam:PassRole` für AWS Glue verfügen.

- Wenn ein Benutzer einen AWS Glue Job erstellt, vergewissern Sie sich, dass die Rolle des Benutzers eine Richtlinie enthält, die `iam:PassRole` für enthält AWS Glue. Weitere Informationen finden Sie unter Schritt 3: Fügen Sie eine Richtlinie an Benutzer an, die auf AWS Glue zugreifen.

Fehler: Die DescribeVpcEndpoints Aktion ist nicht autorisiert. die VPC-ID kann nicht validiert werden vpc-id

- Überprüfen Sie die Richtlinie, an die Sie die Genehmigung übergeben haben. AWS Glue `ec2:DescribeVpcEndpoints`

Fehler: Die DescribeRouteTables Aktion ist nicht autorisiert. Die Subnetz-ID konnte nicht überprüft werden: Subnetz-ID in VPC-ID: vpc-id

- Suchen Sie in der AWS Glue Richtlinie`ec2:DescribeRouteTables`, an die Sie die Genehmigung übergeben haben.

Fehler: ec2 konnte nicht aufgerufen werden: DescribeSubnets

- Überprüfen Sie die Richtlinie, an die Sie AWS Glue die `ec2:DescribeSubnets` Genehmigung übergeben haben.

Fehler: ec2 konnte nicht aufgerufen werden: DescribeSecurityGroups

- Überprüfen Sie die Richtlinie, an die Sie AWS Glue die `ec2:DescribeSecurityGroups` Genehmigung übergeben haben.

Fehler: Subnetz für AZ wurde nicht gefunden

- Die Availability Zone ist möglicherweise nicht verfügbar für AWS Glue. Erstellen und verwenden Sie ein neues Subnetz in einer anderen Availability Zone als der in der Nachricht angegebenen.

Fehler: Ausnahme der Auftragsausführung beim Schreiben in ein JDBC-Ziel

Wenn Sie einen Auftrag ausführen, der in ein JDBC-Ziel schreibt, können bei diesem in den folgenden Szenarien Fehler auftreten:

- Wenn Ihr Auftrag in eine Microsoft SQL Server-Tabelle schreibt und die Tabelle über Spalten verfügt, die als `Boolean`-Typ definiert sind, dann muss die Tabelle in der SQL Server-Datenbank vordefiniert sein. Wenn Sie den Job auf der AWS Glue Konsole mithilfe eines SQL Server-Ziels mit der Option Tabellen in Ihrem Datenziel erstellen definieren, ordnen Sie einer Zielspalte mit Datentyp keine Quellspalten zu`Boolean`. Es kann ein Fehler auftreten, wenn der Auftrag ausgeführt wird.

  Sie können den Fehler vermeiden, indem Sie folgende Schritte ausführen:
  - Wählen Sie eine bestehende Tabelle mit der boolesche Spalte.
  - Bearbeiten Sie die `ApplyMapping`-Transformierung und ordnen Sie die boolesche Spalte in der Quelle einer Zahl oder Zeichenfolge im Ziel zu.
  - Bearbeiten Sie die `ApplyMapping`-Transformation, um die boolesche Spalte aus der Quelle zu entfernen.
- Wenn Ihr Auftrag in eine Oracle-Tabelle schreibt, müssen Sie möglicherweise die Länge der Namen von Oracle-Objekten anpassen. In einigen Versionen von Oracle ist die maximale Länge der ID auf 30 Bytes oder 128 Bytes begrenzt. Diese Größenbeschränkung wirkt sich auf die Tabellennamen und Spaltennamen von Oracle-Zieldatenspeichern aus.

  Sie können den Fehler vermeiden, indem Sie folgende Schritte ausführen:
  - Benennen Sie Oracle-Zieltabellen innerhalb des Limits für Ihre Version.
  - Die Standard-Spaltennamen werden aus den Feldnamen in die Daten generiert. Um mit den Fällen umzugehen, in denen die Spaltennamen länger sind als das Limit, verwenden Sie die `ApplyMapping`- oder `RenameField`-Transformation, um den Namen der Spalte so zu ändern, dass er sich innerhalb des Limits befindet.

Fehler: Amazon S3: Der Vorgang ist für die Speicherklasse des Objekts nicht gültig

Wenn dieser Fehler AWS Glue zurückgegeben wird, hat Ihr AWS Glue Job möglicherweise Daten aus Tabellen gelesen, die Partitionen auf verschiedenen Amazon S3 S3-Speicherklassenstufen haben.

- Durch die Verwendung von Speicherklassenausschlüssen können Sie sicherstellen, dass Ihre AWS Glue Jobs auf Tabellen funktionieren, die über Partitionen in diesen Speicherklassenstufen verfügen. Ohne Ausnahmen schlagen Jobs, die Daten aus diesen Stufen lesen, mit dem folgenden Fehler fehl:. `AmazonS3Exception: The operation is not valid for the object's storage class`

  Weitere Informationen finden Sie unter Ausschließen von Amazon-S3-Speicherklassen.

Fehler: Amazon S3 Zeitüberschreitung

Wenn ein Verbindungs-Timeout-Fehler AWS Glue zurückgegeben wird, kann dies daran liegen, dass versucht wird, auf einen Amazon S3 S3-Bucket in einer anderen AWS Region zuzugreifen.

- Ein Amazon S3 S3-VPC-Endpunkt kann Traffic nur an Buckets innerhalb einer AWS Region weiterleiten. Wenn Sie eine Verbindung mit Buckets in anderen Regionen herstellen müssen, besteht eine mögliche Abhilfe darin, ein NAT-Gateway zu verwenden. Weitere Informationen finden Sie unter NAT-Gateways.

Fehler: Amazon-S3-Zugriff verweigert

Wenn ein Fehler „Zugriff verweigert“ auf einen Amazon S3 S3-Bucket oder ein Amazon S3-Objekt AWS Glue zurückgegeben wird, kann dies daran liegen, dass die angegebene IAM-Rolle keine Richtlinie mit Berechtigungen für Ihren Datenspeicher hat.

- Ein ETL-Auftrag benötigt Zugriff auf einen Amazon-S3-Datenspeicher, der als Quelle oder Ziel verwendet wird. Ein Crawler benötigt Zugriff auf einen Amazon-S3-Datenspeicher, den er durchsucht. Weitere Informationen finden Sie unter Schritt 2: Erstellen einer IAM-Rolle für AWS Glue.

Fehler: Amazon-S3-Zugriffsschlüssel-ID ist nicht vorhanden

Wenn bei der Ausführung eines Jobs der Fehler „Zugriffsschlüssel ID existiert nicht“ AWS Glue zurückgegeben wird, kann dies einen der folgenden Gründe haben:

- Ein ETL-Auftrag verwendet eine IAM-Rolle für den Zugriff auf Datenspeicher, vergewissern Sie sich, dass die IAM-Rolle für Ihren Auftrag nicht gelöscht wurde, bevor der Auftrag gestartet wurde.

- Eine IAM;-Rolle enthält Berechtigungen für den Zugriff auf Ihre Datenspeicher, vergewissern Sie sich, dass alle angefügten Amazon-S3-Richtlinien mit `s3:ListBucket` richtig sind.

Fehler: Auftragsausführung schlägt fehl, wenn auf Amazon S3 mit einem **`s3a://`**-URI zugegriffen wird

Gibt eine Auftragsausführung einen Fehler wie Fehler beim Analysieren des XML-Dokuments mit der Handler-Klasse, zurück, kann dies daran liegen, das versucht wurde, Hunderte Dateien mit dem `s3a://`-URI aufzulisten. Greifen Sie stattdessen mit einem `s3://`-URI auf Ihren Datenspeicher zu. Im folgenden Ausnahme-Trace ist der zu suchende Fehler gekennzeichnet:

```
1. com.amazonaws.SdkClientException: Failed to parse XML document with handler class
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser$ListBucketHandler
2. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseXmlInputStream(XmlRespons
3. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseListBucketObjectsResponse
4. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:70)
5. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:59)
6. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:62)
7. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:31)
8. at
 com.amazonaws.http.response.AwsResponseHandlerAdapter.handle(AwsResponseHandlerAdapter.java:70
9. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.handleResponse(AmazonHttpClient.java:1554)
10. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeOneRequest(AmazonHttpClient.java:1272)
11. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeHelper(AmazonHttpClient.java:1056)
12. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.doExecute(AmazonHttpClient.java:743)
13. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeWithTimer(AmazonHttpClient.java:717)
14. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.execute(AmazonHttpClient.java:699)
```

```
15. at com.amazonaws.http.AmazonHttpClient$RequestExecutor.access
$500(AmazonHttpClient.java:667)
16. at com.amazonaws.http.AmazonHttpClient
$RequestExecutionBuilderImpl.execute(AmazonHttpClient.java:649)
17. at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:513)
18. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4325)
19. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4272)
20. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4266)
21. at com.amazonaws.services.s3.AmazonS3Client.listObjects(AmazonS3Client.java:834)
22. at org.apache.hadoop.fs.s3a.S3AFileSystem.getFileStatus(S3AFileSystem.java:971)
23. at
 org.apache.hadoop.fs.s3a.S3AFileSystem.deleteUnnecessaryFakeDirectories(S3AFileSystem.java:115
24. at org.apache.hadoop.fs.s3a.S3AFileSystem.finishedWrite(S3AFileSystem.java:1144)
25. at org.apache.hadoop.fs.s3a.S3AOutputStream.close(S3AOutputStream.java:142)
26. at org.apache.hadoop.fs.FSDataOutputStream
$PositionCache.close(FSDataOutputStream.java:74)
27. at org.apache.hadoop.fs.FSDataOutputStream.close(FSDataOutputStream.java:108)
28. at org.apache.parquet.hadoop.ParquetFileWriter.end(ParquetFileWriter.java:467)
29. at
 org.apache.parquet.hadoop.InternalParquetRecordWriter.close(InternalParquetRecordWriter.java:1
30. at
 org.apache.parquet.hadoop.ParquetRecordWriter.close(ParquetRecordWriter.java:112)
31. at
 org.apache.spark.sql.execution.datasources.parquet.ParquetOutputWriter.close(ParquetOutputWrit
32. at org.apache.spark.sql.execution.datasources.FileFormatWriter
$SingleDirectoryWriteTask.releaseResources(FileFormatWriter.scala:252)
33. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:191)
34. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:188)
35. at org.apache.spark.util.Utils
$.tryWithSafeFinallyAndFailureCallbacks(Utils.scala:1341)
36. at org.apache.spark.sql.execution.datasources.FileFormatWriter$.org$apache$spark
$sql$execution$datasources$FileFormatWriter$$executeTask(FileFormatWriter.scala:193)
37. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:129)
38. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:128)
39. at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:87)
40. at org.apache.spark.scheduler.Task.run(Task.scala:99)
41. at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:282)
42. at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
```

```
43. at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
44. at java.lang.Thread.run(Thread.java:748)
```

Fehler: Amazon-S3-Servicetoken abgelaufen

Beim Verschieben von Daten von und nach Amazon Redshift werden temporäre Amazon-S3-Anmeldeinformationen, die nach 1 Stunde ablaufen, verwendet. Wenn Sie einen lang laufenden Auftrag haben, kann er fehlschlagen. Informationen zum Einrichten von lang laufenden Aufträgen zum Verschieben von Daten von und zu Amazon Redshift finden Sie unter aws-glue-programming-etl-connect-redshift-home.

Fehler: Keine private DNS für die Netzwerkschnittstelle gefunden

Wenn ein Auftrag fehlschlägt oder ein Entwicklungsendpunkt nicht bereitgestellt werden kann, ist der Grund hierfür möglicherweise ein Problem in der Netzwerkeinrichtung.

- Wenn Sie das von Amazon bereitgestellte DNS verwenden, muss der Wert `enableDnsHostnames` auf "true" gesetzt sein. Weitere Informationen finden Sie unter DNS.

Fehler: Bereitstellung des Entwicklungsendpunkts fehlgeschlagen

Wenn ein Entwicklungsendpunkt AWS Glue nicht erfolgreich bereitgestellt werden kann, kann dies an einem Problem in der Netzwerkkonfiguration liegen.

- Beim Definieren eines Entwicklungsendpunkts werden die VPC, das Subnetz und die Sicherheitsgruppen validiert, um zu bestätigen, dass sie bestimmte Anforderungen erfüllen.
- Wenn Sie den optionalen öffentlichen SSH-Schlüssel bereitgestellt haben, prüfen Sie, ob es sich um einen gültigen öffentlichen SSH-Schlüssel handelt.
- Prüfen Sie in der VPC-Konsole, dass Ihre VPC ein gültiges DHCP-Optionsset verwendet. Weitere Informationen finden Sie unter DHCP-Optionssets.
- Wenn der Cluster im Status PROVISIONING bleibt, wenden Sie sich an den AWS -Support.

Fehler: Notebook-Server-ERSTELLUNG_FEHLGESCHLAGEN

Wenn AWS Glue der Notebook-Server für einen Entwicklungsendpunkt nicht erstellt werden kann, kann dies an einem der folgenden Probleme liegen:

- AWS Glue übergibt eine IAM-Rolle an Amazon EC2 , wenn es den Notebook-Server einrichtet. Die IAM-Rolle muss eine Vertrauensbeziehung zu Amazon EC2 haben.
- Die IAM-Rolle muss ein Instance-Profil mit demselben Namen haben. Wenn Sie die Rolle für Amazon EC2 mit der IAM-Konsole erstellen, wird das Instance-Profil mit dem gleichen Namen automatisch erstellt. Prüfen Sie auf Fehler im Protokoll bezüglich einem ungültigen Instance-Profilnamen `iamInstanceProfile.name`. Weitere Informationen finden Sie unter Instanzprofile verwenden.
- Stellen Sie sicher, dass Ihre Rolle über die Berechtigung für den Zugriff auf `aws-glue*`-Buckets in der Richtlinie verfügt, die Sie zum Erstellen des Notebook-Servers weiterleiten.

Fehler: Lokales Notebook kann nicht gestartet werden

Wenn Ihr lokales Notebook nicht gestartet werden kann und Fehler meldet, dass ein Verzeichnis oder Ordner nicht gefunden werden kann, kann hierfür einer der folgenden Probleme vorliegen:

- Wenn Sie Microsoft Windows ausführen, stellen Sie sicher, dass die `JAVA_HOME`-Umgebungsvariable auf das korrekte Java-Verzeichnis zeigt. Es ist möglich, Java zu aktualisieren, ohne diese Variable zu aktualisieren. Wenn diese auf einen Ordner verweist, der nicht mehr vorhanden ist, können Jupyter Notebooks nicht gestartet werden.

Fehler: Ausführen von Crawler fehlgeschlagen

Wenn AWS Glue ein Crawler zum Katalogisieren Ihrer Daten nicht erfolgreich ausgeführt werden kann, kann das an einem der folgenden Gründe liegen. Prüfen Sie zunächst, ob ein Fehler in der Crawlers-Liste der AWS Glue -Konsole aufgeführt ist. Überprüfen Sie, ob neben dem Crawlernamen ein Ausrufungssymbol vorhanden ist, und fahren Sie mit der Maus über das Symbol, um alle zugehörigen Nachrichten anzuzeigen.

- Suchen Sie in den Protokollen unter Protokolle nach, ob der Crawler ausgeführt wurde CloudWatch . `/aws-glue/crawlers`

Fehler: Partitionen wurden nicht aktualisiert

Falls Ihre Partitionen im Datenkatalog nicht aktualisiert wurden, als Sie einen ETL-Job ausgeführt haben, können diese Protokollanweisungen der `DataSink` Klasse in den CloudWatch Logs hilfreich sein:

- "`Attempting to fast-forward updates to the Catalog - nameSpace:`": gibt an, welche Datenbank, Tabelle und catalogID von diesem Auftrag geändert werden sollen. Wenn diese Anweisung nicht vorhanden ist, prüfen Sie, ob `enableUpdateCatalog` auf "true" gesetzt ist und ordnungsgemäß als `getSink()`-Parameter oder in `additional_options` übergeben wird.
- "`Schema change policy behavior:`": gibt an, welchen `updateBehavior`-Schemawert Sie übergeben haben.
- "`Schemas qualify (schema compare):`": wird den Wert "true" oder "false" annehmen.
- "`Schemas qualify (case-insensitive compare):`": wird den Wert "true" oder "false" annehmen.
- Wenn beide falsch sind und Sie `updateBehavior` nicht auf gesetzt sind `UPDATE_IN_DATABASE`, muss Ihr DynamicFrame Schema identisch sein oder eine Teilmenge der Spalten enthalten, die im Tabellenschema des Datenkatalogs angezeigt werden.

Weitere Informationen zum Aktualisieren von Partitionen finden Sie unter Aktualisierung des Schemas und Hinzufügen neuer Partitionen im Datenkatalog mit AWS Glue ETL-Aufträge.

Fehler: Aktualisierung des Auftragslesezeichens aufgrund einer Versionsabweichung fehlgeschlagen

Möglicherweise versuchen Sie, AWS Glue Jobs so zu parametrieren, dass sie dasselbe transformation/logic on different datasets in Amazon S3. You want to track processed files on the locations provided. When you run the same job on the same source bucket and write to the same/ different Ziel gleichzeitig anwenden (Parallelität >1). Der Job schlägt mit dem folgenden Fehler fehl:

```
py4j.protocol.Py4JJavaError: An error occurred while
 callingz:com.amazonaws.services.glue.util.Job.commit.:com.amazonaws.services.gluejobexecutor.m
 Continuation update failed due to version mismatch. Expected version 2 but found
 version 3
```

Lösung: Setzen Sie die Gleichzeitigkeit auf 1 oder lassen Sie den Auftrag nicht gleichzeitig laufen.

Derzeit unterstützen AWS Glue Lesezeichen keine gleichzeitigen Jobausführungen und Commits schlagen fehl.

Fehler: Ein Auftrag verarbeitet Daten erneut, wenn Auftragslesezeichen aktiviert sind

Es kann vorkommen, dass Sie AWS Glue Job-Lesezeichen aktiviert haben, Ihr ETL-Job jedoch Daten erneut verarbeitet, die bereits in einem früheren Lauf verarbeitet wurden. Überprüfen Sie, ob diese häufigen Ursachen für diesen Fehler vorliegen:

Maximale Parallelität

Wenn Sie die maximale Anzahl gleichzeitiger Ausführungen für den Job auf einen Wert über dem Standardwert 1 festlegen, kann dies zu Problemen mit den Job-Lesezeichen führen. Dies kann der Fall sein, wenn Job-Lesezeichen die Uhrzeit der letzten Änderung von Objekten überprüfen, um zu überprüfen, welche Objekte erneut verarbeitet werden müssen. Weitere Informationen finden Sie in der Beschreibung der maximalen Parallelität unter Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue.

Fehlendes Auftragsobjekt

Stellen Sie sicher, dass Ihr Auftragslaufskript mit dem folgenden Commit endet:

```
job.commit()
```

Wenn Sie dieses Objekt einschließen, AWS Glue zeichnet es den Zeitstempel und den Pfad der Auftragsausführung auf. Wenn Sie den Job erneut mit demselben Pfad ausführen, AWS Glue werden nur die neuen Dateien verarbeitet. Wenn Sie dieses Objekt nicht einbeziehen und Auftragslesezeichen aktiviert sind, verarbeitet der Auftrag die bereits bearbeiteten Dateien zusammen mit den neuen Dateien erneut. Dies führt zu Redundanz im Zieldatenspeicher des Auftrags.

Fehlender Parameter Transformationskontext

Der Transformationskontext ist ein optionaler Parameter in der Klasse `GlueContext`, aber Auftragslesezeichen funktionieren nicht, wenn Sie ihn nicht einbinden. Um diesen Fehler zu beheben, fügen Sie den Transformationskontext-Parameter hinzu, wenn Sie den erstellen DynamicFrame, wie im Folgenden dargestellt:

```
sample_dynF=create_dynamic_frame_from_catalog(database,
 table_name,transformation_ctx="sample_dynF")
```

Eingabequelle

Wenn Sie eine relationale Datenbank (eine JDBC-Verbindung) als Eingabequelle verwenden, funktionieren Auftragslesezeichen nur, wenn die Primärschlüssel der Tabelle in sequentieller Reihenfolge sind. Auftragslesezeichen funktionieren für neue Zeilen, nicht aber für aktualisierte Zeilen. Das liegt daran, dass Auftragslesezeichen nach den Primärschlüsseln suchen, die bereits existieren. Dies gilt nicht, wenn Ihre Eingabequelle Amazon Simple Storage Service (Amazon S3) ist.

Zeitpunkt der letzten Änderung

Für Amazon-S3-Eingabequellen überprüfen Auftragslesezeichen den Zeitpunkt der letzten Änderung der Objekte und nicht die Dateinamen, um zu überprüfen, welche Objekte erneut verarbeitet werden müssen. Wenn Ihre Eingabedaten seit der letzten Auftragsausführung geändert wurden, werden die Dateien bei der erneuten Ausführung des Auftrags erneut verarbeitet.

Fehler: Failover-Verhalten zwischen VPCs AWS Glue

Der folgende Prozess wird für das Failover für Jobs in AWS Glue 4.0 und früheren Versionen verwendet.

Zusammenfassung: Zum Zeitpunkt der Übermittlung einer Auftragsausführung wird eine AWS Glue Verbindung ausgewählt. Wenn bei der Ausführung des Auftrags Probleme auftreten (fehlende IP-Adressen, Verbindung zur Quelle, Routing-Probleme), schlägt die Auftragsausführung fehl. Wenn Wiederholungsversuche konfiguriert sind, AWS Glue wird es mit derselben Verbindung erneut versucht.

1. Prüft bei jedem Ausführungsversuch den Zustand der Verbindungen in der Reihenfolge, AWS Glue die in der Auftragskonfiguration angegeben ist, bis eine Verbindung gefunden wird, die verwendet werden kann. Bei einem Ausfall der Availability Zone (AZ) bestehen die Verbindungen aus dieser AZ die Prüfung nicht und werden übersprungen.
2. AWS Glue validiert die Verbindung mit den folgenden Angaben:
   - prüft, ob eine gültige Amazon-VPC-ID und ein gültiges Subnetz vorhanden sind.
   - prüft, ob ein NAT-Gateway oder ein Amazon-VPC-Endpunkt vorhanden sind.
   - prüft, ob dem Subnetz mehr als 0 zugewiesene IP-Adressen zugewiesen sind.
   - prüft, ob die AZ fehlerfrei ist.

   AWS Glue kann die Konnektivität zum Zeitpunkt der Einreichung der Auftragsausführung nicht überprüfen.

3. Bei Aufträgen, die Amazon VPC verwenden, werden alle Treiber und Ausführer in derselben AZ erstellt, wobei die Verbindung zum Zeitpunkt der Übermittlung der Auftragsausführung ausgewählt wurde.
4. Wenn Wiederholungsversuche konfiguriert sind, AWS Glue wird es mit derselben Verbindung erneut versucht. Dies liegt daran, dass wir nicht garantieren können, dass Probleme mit dieser Verbindung dauerhaft auftreten. Wenn eine AZ ausfällt, können vorhandene Auftragsausführungen (je nach Stadium der Auftragsausführung) in dieser AZ fehlschlagen. Ein erneuter Versuch sollte einen AZ-Fehler erkennen und einen anderen AZ für die neue Ausführung auswählen.

Crawler-Fehler, wenn der Crawler Lake Formation Formation-Berechtigungen verwendet

Verwenden Sie die folgenden Informationen, um verschiedene Probleme bei der Konfiguration des Crawlers mit Lake-Formation-Anmeldeinformationen zu diagnostizieren und zu beheben.

Fehler: Der S3-Standort: s3://examplepath ist nicht registriert

Damit ein Crawler mit Lake-Formation-Anmeldeinformationen ausgeführt werden kann, müssen Sie zuerst Lake Formation-Berechtigungen einrichten. Um diesen Fehler zu beheben, registrieren Sie bitte den Amazon-S3-Zielstandort bei Lake Formation. Weitere Informationen finden Sie unter Registrieren eines Amazon-S3-Speicherorts.

Fehler: User/Role ist nicht autorisiert, Folgendes auszuführen: lakeformation: on resource GetDataAccess

Bitte fügen Sie der Crawler-Rolle die `lakeformation:GetDataAccess`-Berechtigung über die IAM-Konsole oder AWS CLI hinzu. Mit dieser Berechtigung gewährt Lake Formation die Anforderung von temporären Anmeldeinformationen für den Zugriff auf die Daten. Sehen Sie sich die folgende Richtlinie an:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": [
      "lakeformation:GetDataAccess"
    ],
    "Resource": "*"
  }
}
```

Fehler: Unzureichende Lake-Formation-Berechtigung(en) für (Datenbankname: exampleDatabase, Tabellenname: exampleTable)

Erteilen Sie der Crawlerrolle in der Lake Formation Formation-Konsole (https://console.aws.amazon.com/lakeformation/) Zugriffsberechtigungen (`CreateDescribe`,,`Alter`) für die Datenbank, die als Ausgabedatenbank angegeben ist. Sie können auch Berechtigungen für die Tabelle erteilen. Weitere Informationen finden Sie unter Erteilen von Datenbankberechtigungen mit der benannten Ressourcenmethode.

Fehler: Unzureichende Lake-Formation-Berechtigung(en) auf s3://examplepath

1. Kontoübergreifendes Crawling
   a. Melden Sie sich bei der Lake Formation Formation-Konsole (https://console.aws.amazon.com/lakeformation/) mit dem Konto an, in dem der Amazon S3 S3-Bucket registriert ist (Konto B). Erteilen Sie dem Konto, in dem der Crawler ausgeführt werden soll, die Berechtigungen für den Datenstandort. Dadurch darf der Crawler Daten vom Amazon-S3-Zielstandort lesen.
   b. Erteilen Sie in dem Konto, in dem der Crawler erstellt wird (Konto A), der für die Ausführung des Crawlers verwendeten IAM-Rolle die Berechtigungen für den Datenstandort am Amazon-S3-Zielstandort, damit der Crawler die Daten vom Ziel in Lake Formation lesen kann. Weitere Informationen finden Sie unter Erteilen von Berechtigungen für den Datenstandort (externes Konto).
2. Crawling im Konto (Crawler und registrierter Amazon-S3-Speicherort befinden sich im selben Konto) – Erteilen Sie der IAM-Rolle, die für die Ausführung des Crawlers auf dem Amazon-S3-Standort verwendet wird, die Berechtigungen für den Datenstandort, damit der Crawler die Daten aus dem Zielstandort in Lake Formation lesen kann. Weitere Informationen finden Sie unter Erteilen von Datenspeicherortberechtigungen (im selben Konto).

Häufig gestellte Fragen zur Crawler-Konfiguration mit Lake-Formation-Anmeldeinformationen

1. Wie konfiguriere ich einen Crawler für die Ausführung mit Lake-Formation-Anmeldeinformationen mit der AWS -Konsole?

   Wählen Sie in der AWS Glue Konsole (https://console.aws.amazon.com/glue/) bei der Konfiguration des Crawlers die Option Lake Formation Formation-Anmeldeinformationen für das Crawlen der Amazon S3 S3-Datenquelle verwenden aus. Geben Sie für kontoübergreifendes Crawling die AWS-Konto ID an, unter der der Amazon S3 S3-Zielstandort bei Lake Formation registriert ist. Das Feld accountId (Konto-ID) ist optional für das Crawling im Konto.

2. Wie konfiguriere ich einen Crawler so, dass er mit Lake-Formation-Anmeldeinformationen mit AWS CLI ausgeführt wird?

   Fügen Sie während des `CreateCrawler`-API-Aufrufs `LakeFormationConfiguration` hinzu:

   ```
   "LakeFormationConfiguration": {
       "UseLakeFormationCredentials": true,
       "AccountId": "111111111111" (AWS account ID where the target Amazon S3 location
    is registered with Lake Formation)
     }
   ```

3. Was sind die unterstützten Ziele für einen Crawler, der Lake-Formation-Anmeldeinformationen verwendet?

   Ein Crawler, der Lake-Formation-Anmeldeinformationen verwendet, wird nur für Amazon S3 (kontointernes und kontoübergreifendes Crawling), kontointerne Data-Catalog-Ziele (wobei der zugrunde liegende Speicherort Amazon S3 ist) und Apache-Iceberg-Ziele unterstützt.

4. Kann ich mehrere Amazon-S3-Buckets als Teil eines einzelnen Crawlers mit Lake-Formation-Anmeldeinformationen crawlen?

   Nein, für Crawling-Ziele, die Lake-Formation-Berechtigungsverkauf verwenden, müssen die zugrunde liegenden Amazon-S3-Standorte zum selben Bucket gehören. Kunden können beispielsweise mehrere Zielstandorte (`s3://bucket1/folder1, s3://bucket1/folder2`) verwenden, wenn diese sich im selben Bucket befinden (Bucket1). Das Angeben verschiedener Buckets (s3://bucket1/folder1, s3://bucket2/folder2) wird nicht unterstützt.

Fehlerbehebung AWS Glue bei Ray-Fehlern aus Protokollen

AWS Glue bietet Zugriff auf Protokolle, die von Ray-Prozessen während der Jobausführung ausgegeben werden. Wenn Sie bei Ray-Aufträgen auf Fehler oder unerwartetes Verhalten stoßen, sammeln Sie zunächst Informationen aus den Protokollen, um die Fehlerursache zu ermitteln. Wir stellen auch ähnliche Protokolle für interaktive Sitzungen bereit. Sitzungsprotokolle werden mit dem `/aws-glue/ray/sessions`-Präfix bereitgestellt.

Protokollzeilen werden CloudWatch in Echtzeit gesendet, während Ihr Job ausgeführt wird. Nach Abschluss der Ausführung werden die Druckanweisungen an die CloudWatch Protokolle angehängt. Protokolle werden nach der Ausführung eines Auftrags zwei Wochen lang aufbewahrt.

Überprüfung von Ray-Auftragsprotokollen

Wenn ein Auftrag fehlschlägt, erfassen Sie den Auftragsnamen und die ID der Auftragsausführung. Sie finden diese in der AWS Glue Konsole. Navigieren Sie zur Seite des Auftrags und dann zur Registerkarte Runs (Ausführungen). Ray-Job-Protokolle werden in den folgenden dedizierten CloudWatch Protokollgruppen gespeichert.

- `/aws-glue/ray/jobs/script-log/` – Speichert Protokolle, die von Ihrem Haupt-Ray-Skript ausgegeben werden.
- `/aws-glue/ray/jobs/ray-monitor-log/` – Speichert Protokolle, die vom Ray-Autoscaler-Prozess ausgegeben werden. Diese Protokolle werden für den Hauptknoten und nicht für andere Worker-Knoten generiert.
- `/aws-glue/ray/jobs/ray-gcs-logs/` – Speichert Protokolle, die vom GCS-Prozess (Global Control Store) ausgegeben werden. Diese Protokolle werden für den Hauptknoten und nicht für andere Worker-Knoten generiert.
- `/aws-glue/ray/jobs/ray-process-logs/` – Speichert Protokolle, die von anderen Ray-Prozessen (hauptsächlich dem Dashboard-Agenten) ausgegeben werden, die auf dem Hauptknoten ausgeführt werden. Diese Protokolle werden für den Hauptknoten und nicht für andere Worker-Knoten generiert.
- `/aws-glue/ray/jobs/ray-raylet-logs/` – Speichert Protokolle, die von jedem Raylet-Prozess ausgegeben werden. Diese Protokolle werden in einem einzigen Stream für jeden Worker-Knoten erfasst, einschließlich des Hauptknotens.
- `/aws-glue/ray/jobs/ray-worker-out-logs/` – Speichert `stdout`-Protokolle für jeden Worker im Cluster. Diese Protokolle werden für jeden Worker-Knoten generiert, einschließlich des Hauptknotens.

- `/aws-glue/ray/jobs/ray-worker-err-logs/` – Speichert `stderr`-Protokolle für jeden Worker im Cluster. Diese Protokolle werden für jeden Worker-Knoten generiert, einschließlich des Hauptknotens.
- `/aws-glue/ray/jobs/ray-runtime-env-log/` – Speichert Protokolle über den Ray-Einrichtungsprozess. Diese Protokolle werden für jeden Worker-Knoten generiert, einschließlich des Hauptknotens.

Fehlerbehebung bei Ray-Auftragsfehlern

Um die Organisation von Ray-Protokollgruppen zu verstehen und die Protokollgruppen zu finden, die Ihnen bei der Fehlerbehebung helfen, ist es hilfreich, Hintergrundinformationen zur Ray-Architektur zu haben.

In AWS Glue ETL entspricht ein Worker einer Instanz. Wenn Sie Worker für einen AWS Glue Job konfigurieren, legen Sie die Art und Anzahl der Instanzen fest, die für den Job reserviert sind. Ray verwendet den Begriff Worker auf unterschiedliche Weise.

Ray verwendet Head-Knoten und Worker-Knoten, um die Verantwortlichkeiten einer Instance innerhalb eines Ray-Clusters zu unterscheiden. Ein Ray-Worker-Knoten kann mehrere Akteurs-Prozesse hosten, die Berechnungen durchführen, um das Ergebnis Ihrer verteilten Berechnung zu erzielen. Akteure, die eine Replik einer Funktion ausführen, werden als Repliken bezeichnet. Replikatakteure können auch als Worker-Prozesse bezeichnet werden. Replikate können auch auf dem Hauptknoten ausgeführt werden, der als Head bezeichnet wird, da er zusätzliche Prozesse zur Koordinierung des Clusters ausführt.

Jeder Akteur, der zu Ihrer Berechnung beiträgt, generiert einen eigenen Protokollstream. Das gibt uns einige Einblicke:

- Die Anzahl der Prozesse, die Protokolle ausgeben, kann größer sein als die Anzahl der Worker, die dem Auftrag zugewiesen sind. Oft hat jeder Kern auf jeder Instance einen Akteur.
- Ray-Head-Knoten geben Cluster-Management- und Startprotokolle aus. Im Gegensatz dazu geben Ray-Worker-Knoten nur Protokolle für die an ihnen ausgeführten Arbeiten aus.

Weitere Informationen zur Ray-Architektur finden Sie in den Architektur-Whitepapers in der Ray-Dokumentation.

Problembereich: Zugriff auf Amazon S3

Prüfen Sie die Fehlermeldung der Auftragsausführung. Wenn dies nicht genug Informationen enthält, lesen Sie `/aws-glue/ray/jobs/script-log/`.

Problembereich: PIP-Abhängigkeitsmanagement

Überprüfen Sie `/aws-glue/ray/jobs/ray-runtime-env-log/`.

Problembereich: Überprüfung von Zwischenwerten im Hauptprozess

Schreiben Sie von Ihrem Hauptskript nach `stderr` oder `stdout` und rufen Sie Protokolle von `/aws-glue/ray/jobs/script-log/` ab.

Problembereich: Überprüfung von Zwischenwerten in einem untergeordneten Prozess

Schreiben Sie nach `stderr` oder `stdout` von Ihrer `remote`-Funktion. Rufen Sie anschließend Protokolle von `/aws-glue/ray/jobs/ray-worker-out-logs/` oder `/aws-glue/ray/jobs/ray-worker-err-logs/` ab. Ihre Funktion wurde möglicherweise auf einem beliebigen Replikat ausgeführt, sodass Sie möglicherweise mehrere Protokolle untersuchen müssen, um die beabsichtigte Ausgabe zu finden.

Problembereich: Interpretation von IP-Adressen in Fehlermeldungen

In bestimmten Fehlersituationen kann Ihr Auftrag möglicherweise eine Fehlermeldung ausgeben, die eine IP-Adresse enthält. Diese IP-Adressen sind flüchtig und werden vom Cluster zur Identifizierung und Kommunikation zwischen den Knoten verwendet. Protokolle für einen Knoten werden in einem Protokollstream mit einem eindeutigen Suffix basierend auf der IP-Adresse veröffentlicht.

In können Sie Ihre Protokolle filtern CloudWatch, um die für diese IP-Adresse spezifischen Protokolle zu überprüfen, indem Sie dieses Suffix identifizieren. Wenn Sie beispielsweise *`FAILED_IP`* und *`JOB_RUN_ID`* angeben, können Sie das Suffix wie folgt identifizieren:

```
filter @logStream like /JOB_RUN_ID/
| filter @message like /IP-/
| parse @message "IP-[*]" as ip
| filter ip like /FAILED_IP/
| fields replace(ip, ":", "_") as uIP
| stats count_distinct by uIP as logStreamSuffix
| display logStreamSuffix
```

AWS Glue Ausnahmen für maschinelles Lernen

In diesem Thema werden HTTP-Fehlercodes und Zeichenfolgen für beschrieben AWS Glue Ausnahmen im Zusammenhang mit maschinellem Lernen. Die Fehlercodes und Fehlerzeichenfolgen werden für jede Machine Learning-Aktivität bereitgestellt, die beim Ausführen einer Operation auftreten kann. Außerdem können Sie sehen, ob es möglich ist, die Operation, die zu dem Fehler geführt hat, erneut zu versuchen.

Stornieren MLTask RunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es wurde kein ML-Aufgabenlauf für [taskRunId]: im Konto [accountId] für die Transformation [TransformName] gefunden.“

  Erneuter Versuch OK: Nein.

Erstellen MLTask RunActivity

Diese Aktivität hat folgende Ausnahmen:

- InvalidInputException (400)
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“
  - „Ein AWS Glue Die Tabelleneingabequelle sollte in der Transformation angegeben werden.“
  - „In der Eingabequellspalte [columnName] ist ein ungültiger Datentyp im Katalog definiert.“
  - „Es muss genau eine Eingabe-Datensatztabelle bereitgestellt werden.“
  - „Datenbankname sollte angegeben werden.“
  - „Tabellenname sollte angegeben werden.“
  - „Schema ist nicht für die Transformation definiert.“
  - „Schema sollte den angegebenen Primärschlüssel enthalten: primaryKey].“
  - „Problem beim Abrufen des Datenkatalogschemas: [message].“

- „Max. Kapazität und Worker-Num/Typ können nicht gleichzeitig eingestellt werden.“
- „Beides WorkerType und NumberOfWorkers sollte gesetzt werden.“
- „MaxCapacity sollte >= [maxCapacity] sein.“
- „NumberOfWorkers sollte >= [MaxCapacity] sein.“
- „Max. Wiederholungen sollten nicht negativ sein.“
- „Parameter für Übereinstimmungssuche wurden nicht festgelegt.“
- „Ein Primärschlüssel muss in den Parametern für die Übereinstimmungssuche angegeben werden.

Erneuter Versuch OK: Nein.

- AlreadyExistsException (400)
  - „Transformation mit dem Namen [TransformName] ist bereits vorhanden.“

Erneuter Versuch OK: Nein.

- IdempotentParameterMismatchException (400)
  - „Idempotente Erstellungsanforderung für Transformation [transformName] hatte nicht übereinstimmende Parameter.“

Erneuter Versuch OK: Nein.

- InternalServiceException (500)
  - „Abhängigkeitsfehler.“

Erneuter Versuch OK: Ja.

- ResourceNumberLimitExceededException (400)
  - „Die Anzahl der ML-Transformationen ([count]) hat die Grenze von [limit] Transformationen überschritten.“

Erneuter Versuch OK: Ja, sobald Sie eine Transformation gelöscht haben, um Platz für diese neue zu schaffen.

MLTransformAktivität löschen

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)

- „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden“

Erneuter Versuch OK: Nein.

Holen MLTask RunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es wurde kein ML-Aufgabenlauf für [taskRunId]: im Konto [accountId] für die Transformation [TransformName] gefunden.“

Erneuter Versuch OK: Nein.

Holen MLTask RunsActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es wurde kein ML-Aufgabenlauf für [taskRunId]: im Konto [accountId] für die Transformation [TransformName] gefunden.“

Erneuter Versuch OK: Nein.

Aktivität abrufen MLTransform

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

Erneuter Versuch OK: Nein.

Aktivität abrufen MLTransforms

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Konto-ID darf nicht leer sein.“
  - „Sortieren für Spalte [column] nicht unterstützt.“
  - „[column] darf nicht leer sein.“
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“

  Erneuter Versuch OK: Nein.

GetSaveLocationForTransformArtifactActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Nicht unterstützter Artefakttyp [artifactType].“
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“

  Erneuter Versuch OK: Nein.

GetTaskRunArtifactActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es wurde kein ML-Aufgabenlauf für [taskRunId]: im Konto [accountId] für die Transformation [TransformName] gefunden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Dateiname '[fileName]' ist für die Veröffentlichung ungültig.“
  - „Artefakt für den Aufgabentyp [taskType] kann nicht abgerufen werden.“
  - „Artefakt für [artifactType] kann nicht abgerufen werden.“
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“

  Erneuter Versuch OK: Nein.

Veröffentlichen MLTransform ModelActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Ein vorhandenes Modell mit Version - [version] kann nicht für Konto-ID - [accountId] - und Transform-ID - [transformId] gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Dateiname '[fileName]' ist für die Veröffentlichung ungültig.“
  - „Unzulässiges vorangestelltes Minuszeichen für vorzeichenlose Zeichenfolge [string].“
  - „UngültigeZiffer am Ende von [string].“
  - „Zeichenfolgenwert [string] überschreitet den Bereich von ‚vorzeichenlos lang‘.“

- „Interner Servicefehler aufgrund unerwarteter Eingaben.“

Erneuter Versuch OK: Nein.

PullLatestMLTransformModelActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“

  Erneuter Versuch OK: Nein.
- ConcurrentModificationException (400)
  - „Es kann keine Modellversion zum Schulen aufgrund von Racing Inserts mit nicht übereinstimmenden Parametern erstellt werden.“
  - „Das ML-Transform-Modell für die Transform-ID [transformId] ist veraltet oder wird von einem anderen Prozess aktualisiert. Versuchen Sie es erneut.“

  Erneuter Versuch OK: Ja.

PutJobMetadataForMLTransformAktivität

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es wurde kein ML-Aufgabenlauf für [taskRunId]: im Konto [accountId] für die Transformation [TransformName] gefunden.“

  Erneuter Versuch OK: Nein.

- InvalidInputException (400)
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“
  - „Unbekannter Aufgabenmetadatentyp [jobType].“
  - „Zum Aktualisieren muss eine Aufgaben-Run-ID angegeben werden.“

  Erneuter Versuch OK: Nein.

StartExportLabelsTaskRunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“
  - „Es existiert kein Labelset für TransformID [transformId] in Konto-ID [accountId].“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - “[message].”
  - „Der bereitgestellte S3-Pfad befindet sich nicht in der gleichen Region wie die Transformation. Erwartete Region - [region], jedoch tatsächlich - [region].“

  Erneuter Versuch OK: Nein.

StartImportLabelsTaskRunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - “[message].”
  - „Ungültiger Labeldateipfad.“

- „Auf die Labeldatei unter [labelPath] kann nicht zugegriffen werden. [message].“
- „Die in der Transformation bereitgestellte IAM-Rolle kann nicht verwendet werden. Rolle: [role].“
- „Ungültige Labeldatei der Größe 0.“
- „Der bereitgestellte S3-Pfad befindet sich nicht in der gleichen Region wie die Transformation. Erwartete Region - [region], jedoch tatsächlich - [region].“

Erneuter Versuch OK: Nein.

- ResourceNumberLimitExceededException (400)
  - „Die Labeldatei hat das Limit von [limit] MB überschritten.“

  Erneuter Versuch OK: Nein. Erwägen Sie, Ihre Labeldatei in mehrere kleinere Dateien zu unterteilen.

Anfang MLEvaluation TaskRunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Es muss genau eine Eingabe-Datensatztabelle bereitgestellt werden.“
  - „Datenbankname sollte angegeben werden.“
  - „Tabellenname sollte angegeben werden.“
  - „Parameter für Übereinstimmungssuche wurden nicht festgelegt.“
  - „Ein Primärschlüssel muss in den Parametern für die Übereinstimmungssuche angegeben werden.

  Erneuter Versuch OK: Nein.
- MLTransformNotReadyException (400)
  - „Diese Operation kann nur auf eine Transformation angewendet werden, die sich im Status READY befindet.“

Erneuter Versuch OK: Nein.

- InternalServiceException (500)
  - „Abhängigkeitsfehler.“

  Erneuter Versuch OK: Ja.
- ConcurrentRunsExceededException (400)
  - „Anzahl der ML-Aufgabenausführungen [count] hat das Transformationslimit von [limit] Aufgabenausführungen überschritten.“
  - „Anzahl der ML-Aufgabenausführungen [count] hat das Limit von [limit] Aufgabenausführungen überschritten.“

  Erneuter Versuch OK: Ja, nach dem Warten auf den Abschluss der Aufgabenausführungen.

Anfang MLLabeling SetGenerationTaskRunActivity

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Es muss genau eine Eingabe-Datensatztabelle bereitgestellt werden.“
  - „Datenbankname sollte angegeben werden.“
  - „Tabellenname sollte angegeben werden.“
  - „Parameter für Übereinstimmungssuche wurden nicht festgelegt.“
  - „Ein Primärschlüssel muss in den Parametern für die Übereinstimmungssuche angegeben werden.

  Erneuter Versuch OK: Nein.
- InternalServiceException (500)
  - „Abhängigkeitsfehler.“

  Erneuter Versuch OK: Ja.

- ConcurrentRunsExceededException (400)
  - „Anzahl der ML-Aufgabenausführungen [count] hat das Transformationslimit von [limit] Aufgabenausführungen überschritten.“

  Erneuter Versuch OK: Ja, nach dem Abschluss der Aufgabenausführungen.

MLTransformAktivität aktualisieren

Diese Aktivität hat folgende Ausnahmen:

- EntityNotFoundException (400)
  - „Kann MLTransform im Konto [AccountID] mit dem Handle [TransformName] nicht gefunden werden.“

  Erneuter Versuch OK: Nein.
- InvalidInputException (400)
  - „Eine weitere Transformation mit dem Namen [transformName] existiert bereits.“
  - “[message].”
  - „Transformationsname darf nicht leer sein.“
  - „Max. Kapazität und Worker-Num/Typ können nicht gleichzeitig eingestellt werden.“
  - „Beides WorkerType und NumberOfWorkers sollte eingestellt werden.“
  - „MaxCapacity sollte >= [minMaxCapacity] sein.“
  - „NumberOfWorkers sollte >= [minNumWorkers] sein.“
  - „Max. Wiederholungen sollten nicht negativ sein.“
  - „Interner Servicefehler aufgrund unerwarteter Eingaben.“
  - „Parameter für Übereinstimmungssuche wurden nicht festgelegt.“
  - „Ein Primärschlüssel muss in den Parametern für die Übereinstimmungssuche angegeben werden.

  Erneuter Versuch OK: Nein.
- AlreadyExistsException (400)
  - „Transformation mit dem Namen [TransformName] ist bereits vorhanden.“

  Erneuter Versuch OK: Nein.
- IdempotentParameterMismatchException (400)

- „Idempotente Erstellungsanforderung für Transformation [transformName] hatte nicht übereinstimmende Parameter.“

Erneuter Versuch OK: Nein.

AWS Glue Kontingente

Sie können sich an uns wenden AWS -Support , um eine Erhöhung des Kontingents für die in der aufgeführten Servicekontingente zu beantragen Allgemeine AWS-Referenz. Wenn nicht anders angegeben, gilt jedes Kontingent spezifisch für eine Region. Weitere Informationen finden Sie unter AWS Glue Endpunkte und Kontingente.

Verbesserung der AWS Glue Leistung

Basisstrategie für die Leistungsoptimierung

Um die AWS Glue Leistung zu verbessern, können Sie erwägen, bestimmte leistungsbezogene AWS Glue Parameter zu aktualisieren. Wenden Sie die folgenden bewährten Methoden an, wenn Sie das Optimieren von Parametern vorbereiten:

- Legen Sie Ihre Leistungsziele fest, bevor Sie beginnen, Probleme zu identifizieren.
- Identifizieren Sie Probleme anhand von Metriken, bevor Sie versuchen, die Optimierungsparameter zu ändern.

Damit Sie beim Optimieren eines Auftrags möglichst konsistente Ergebnisse erzielen, sollten Sie eine Basisstrategie für Ihre Optimierungsarbeit entwickeln.

Im Allgemeinen wird die Leistungsoptimierung im folgenden Workflow durchgeführt:

1. Leistungsziele ermitteln.
2. Metriken messen.
3. Engpässe identifizieren.
4. Die Auswirkung von Engpässen verringern.
5. Wiederholen Sie die Schritte 2 bis 4, bis Sie das beabsichtigte Ziel erreicht haben.

Optimierte Strategien für Ihren Jobtyp

Spark-Jobs — folgen Sie den Anleitungen unter Bewährte Methoden zur Leistungsoptimierung AWS Glue für Apache Spark-Jobs auf AWS Prescriptive Guidance.

Andere Jobs — Sie können Ray- und AWS Glue Python-Shell-Jobs anpassen AWS Glue , indem Sie Strategien anpassen, die in anderen Laufzeitumgebungen verfügbar sind.

Verbesserung der Leistung AWS Glue für Apache Spark-Jobs

Um die Leistung von Spark AWS Glue zu verbessern, können Sie erwägen, bestimmte leistungsbezogene AWS Glue und Spark-Parameter zu aktualisieren.

Weitere Informationen zu spezifischen Strategien zur Identifizierung von Engpässen anhand von Metriken und zur Reduzierung ihrer Auswirkungen finden Sie unter Bewährte Methoden zur Leistungsoptimierung AWS Glue für Apache Spark-Jobs auf AWS Prescriptive Guidance. In diesem Handbuch werden wichtige Themen vorgestellt, die für Apache Spark in allen Laufzeitumgebungen relevant sind, z. B. die Spark-Architektur und Resilient Distributed Datasets. Anhand dieser Themen hilft Ihnen der Leitfaden bei der Implementierung bestimmter Strategien zur Leistungsoptimierung, wie z. B. der Optimierung von Shuffles und der Parallelisierung von Aufgaben.

Sie können Engpässe erkennen, indem Sie die Konfiguration so konfigurieren AWS Glue , dass die Spark-Benutzeroberfläche angezeigt wird. Weitere Informationen finden Sie unter the section called “Überwachen über die Spark-Benutzeroberfläche”.

Darüber hinaus AWS Glue bietet es Leistungsmerkmale, die möglicherweise für den spezifischen Datenspeichertyp gelten, mit dem Ihr Job eine Verbindung herstellt. Referenzinformationen zu Leistungsparametern für Datenspeicher finden Sie unterthe section called “Verbindungsparameter”.

Optimierung von Lesevorgängen mit Pushdown in AWS Glue ETL

Pushdown ist eine Optimierungstechnik, die die Logik zum Abrufen von Daten näher an die Quelle Ihrer Daten verschiebt. Die Quelle könnte eine Datenbank oder ein Dateisystem wie Amazon S3 sein. Wenn Sie bestimmte Operationen direkt an der Quelle ausführen, können Sie Zeit und Rechenleistung sparen, indem Sie nicht alle Daten über das Netzwerk an die von AWS Glue verwaltete Spark-Engine übertragen.

Anders ausgedrückt: Der Pushdown verringert die Menge an gescannten Daten. Weitere Informationen darüber, wie Sie feststellen können, wann diese Technik geeignet ist, finden Sie unter Reduce the amount of data scan im Leitfaden Best Practices for Performance Tuning AWS Glue for Apache Spark Jobs auf AWS Prescriptive Guidance.

Prädikat-Pushdown für auf Amazon S3 gespeicherte Dateien

Wenn Sie mit Dateien auf Amazon S3 arbeiten, die nach Präfixen organisiert wurden, können Sie Ihre Amazon-S3-Zielpfade filtern, indem Sie ein Pushdown-Prädikat definieren. Anstatt den vollständigen Datensatz zu lesen und Filter innerhalb eines `DynamicFrame` anzuwenden, können Sie den Filter direkt auf die im AWS Glue Data Catalog gespeicherten Partitionsmetadaten anwenden. Mit diesem Ansatz können Sie selektiv nur die erforderlichen Daten auflisten und lesen. Weitere Informationen zu diesem Prozess, einschließlich des Schreibens in einen Bucket nach Partitionen, finden Sie unter the section called “Verwalten von Partitionen”.

Mithilfe des `push_down_predicate`-Parameters erreichen Sie einen Prädikat-Pushdown in Amazon S3. Betrachten Sie einen Bucket in Amazon S3, den Sie nach Jahr, Monat und Tag partitioniert haben. Wenn Sie Kundendaten für Juni 2022 abrufen möchten, können Sie AWS Glue anweisen, nur relevante Amazon S3 S3-Pfade zu lesen. Das `push_down_predicate` ist in diesem Fall `year='2022' and month='06'`. Zusammengenommen kann der Lesevorgang wie folgt durchgeführt werden:

Python

```
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database = "customer_db",
    table_name = "customer_tbl",
    push_down_predicate = "year='2022' and month='06'"
)
```

Scala

```
val customer_records = glueContext.getCatalogSource(
database="customer_db",
tableName="customer_tbl",
pushDownPredicate="year='2022' and month='06'"
).getDynamicFrame()
```

Im vorherigen Szenario ruft `push_down_predicate` eine Liste aller Partitionen aus dem AWS Glue Data Catalog ab und filtert sie, bevor die zugrunde liegenden Amazon-S3-Dateien gelesen werden. Obwohl dies in den meisten Fällen hilfreich ist, kann das Auflisten von Partitionen bei der Arbeit mit Datensätzen mit Millionen von Partitionen zeitaufwändig sein. Um dieses Problem zu beheben, kann die serverseitige Bereinigung von Partitionen zur Verbesserung der Leistung eingesetzt werden. Dies geschieht, indem ein Partitionsindex für Ihre Daten im AWS Glue-Datenkatalog erstellt wird. Weitere Informationen zu Partitionsindizes finden Sie unter the section called “Partitionsindizes erstellen”. Anschließend können Sie die `catalogPartitionPredicate`-Option verwenden, um auf den Index zu verweisen. Ein Beispiel zum Abrufen von Partitionen mit `catalogPartitionPredicate` finden Sie unter the section called “Katalogpartitionsprädikate”.

Pushdown beim Arbeiten mit JDBC-Quellen

Der im verwendete AWS Glue-JDBC-Reader `GlueContext` unterstützt Pushdown für unterstützte Datenbanken, indem er benutzerdefinierte SQL-Abfragen bereitstellt, die direkt auf der Quelle ausgeführt werden können. Dies kann durch Einstellen des `sampleQuery`-Parameters erreicht

werden. Ihre Beispielabfrage kann angeben, welche Spalten ausgewählt werden sollen, und ein Pushdown-Prädikat bereitstellen, um die an die Spark-Engine übertragenen Daten zu begrenzen.

Standardmäßig werden Beispielabfragen auf einem einzelnen Knoten ausgeführt, was bei der Verarbeitung großer Datenmengen zu Auftragsfehlern führen kann. Um dieses Feature zur Abfrage von Daten in großem Umfang zu verwenden, sollten Sie die Abfragepartitionierung konfigurieren, indem Sie `enablePartitioningForSampleQuery` auf wahr festlegen. Dadurch wird die Abfrage auf mehrere Knoten über einen von Ihnen gewählten Schlüssel verteilt. Für die Abfragepartitionierung sind außerdem einige weitere notwendige Konfigurationsparameter erforderlich. Weitere Informationen zur Abfragepartitionierung finden Sie unter the section called “Parallel aus JDBC lesen”.

Bei der Einstellung `enablePartitioningForSampleQuery` kombiniert AWS Glue bei der Abfrage Ihrer Datenbank Ihr Pushdown-Prädikat mit einem Partitionierungsprädikat. Sie `sampleQuery` müssen mit einem `AND` für AWS Glue enden, um Partitionierungsbedingungen anzuhängen. (Wenn Sie kein Pushdown-Prädikat angeben, muss `sampleQuery` mit einem `WHERE` enden). Unten sehen Sie ein Beispiel, in dem wir ein Prädikat nach unten verschieben, um nur Zeilen abzurufen, deren `id` größer ist als 1000. Dieser `sampleQuery` gibt nur die Namens- und Standortspalten für Zeilen zurück, in denen `id` größer als der angegebene Wert ist:

Python

```
sample_query = "select name, location from customer_tbl WHERE id>=1000 AND"
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database="customer_db",
    table_name="customer_tbl",
    sample_query = "select name, location from customer_tbl WHERE id>=1000 AND",

    additional_options = {
                           "hashpartitions": 36 ,
                           "hashfield":"id",
                           "enablePartitioningForSampleQuery":True,
                           "sampleQuery":sample_query
                          }
)
```

Scala

```
val additionalOptions = Map(
        "hashpartitions" -> "36",
        "hashfield" -> "id",
```

```
        "enablePartitioningForSampleQuery" -> "true",
        "sampleQuery" -> "select name, location from customer_tbl WHERE id >= 1000
 AND"
        )

    val customer_records = glueContext.getCatalogSource(
        database="customer_db",
        tableName="customer_tbl").getDynamicFrame()
```

> **Note**
>
> Wenn es in Ihrem Datenkatalog und dem zugrunde liegenden Datenspeicher einen anderen Namen `customer_tbl` hat, müssen Sie den Namen der zugrunde liegenden Tabelle in sample_query angeben, da die Abfrage an den zugrunde liegenden Datenspeicher übergeben wird.

Sie können auch JDBC-Tabellen abfragen, ohne sie in den AWS Glue-Datenkatalog zu integrieren. Anstatt Benutzername und Passwort als Parameter für die Methode anzugeben, können Sie Anmeldeinformationen einer bereits vorhandenen Verbindung wiederverwenden, indem Sie `useConnectionProperties` und `connectionName` angeben. In diesem Beispiel rufen wir Anmeldeinformationen von einer Verbindung mit dem Namen `my_postgre_connection` ab.

Python

```
connection_options_dict = {
    "useConnectionProperties": True,
    "connectionName": "my_postgre_connection",
    "dbtable":"customer_tbl",
    "sampleQuery":"select name, location from customer_tbl WHERE id>=1000 AND",
    "enablePartitioningForSampleQuery":True,
    "hashfield":"id",
    "hashpartitions":36
    }

customer_records = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options=connection_options_dict
    )
```

Scala

```
val connectionOptionsJson = """
      {
        "useConnectionProperties": true,
        "connectionName": "my_postgre_connection",
        "dbtable": "customer_tbl",
        "sampleQuery": "select name, location from customer_tbl WHERE id>=1000 AND",
        "enablePartitioningForSampleQuery" : true,
        "hashfield" : "id",
        "hashpartitions" : 36
      }
    """

    val connectionOptions = new JsonOptions(connectionOptionsJson)

    val dyf = glueContext.getSource("postgresql",
 connectionOptions).getDynamicFrame()
```

Hinweise und Einschränkungen für Pushdown in AWS Glue

Das Konzept von Pushdown ist anwendbar, wenn aus Quellen gelesen wird, die nicht gestreamt werden. AWS Glue unterstützt eine Vielzahl von Quellen — die Fähigkeit zum Drücken hängt von der Quelle und dem Anschluss ab.

- Beim Herstellen einer Verbindung mit Snowflake können Sie die Option `query` verwenden. Ähnliche Funktionen gibt es im Redshift-Konnektor in AWS Glue 4.0 und späteren Versionen. Weitere Informationen zum Lesen aus Snowflake mit `query` finden Sie unter the section called “Aus Snowflake lesen”.
- Der DynamoDB-ETL-Reader unterstützt keine Filter oder Pushdown-Prädikate. MongoDB und DocumentDB unterstützen diese Art von Funktionalität ebenfalls nicht.
- Beim Lesen von in Amazon S3 gespeicherten Daten in offenen Tabellenformaten reicht die Partitionierungsmethode für Dateien in Amazon S3 nicht mehr aus. Informationen zum Lesen und Schreiben von Partitionen mithilfe offener Tabellenformate finden Sie in der Dokumentation zum Format.
- DynamicFrame Methoden führen keinen Amazon S3 S3-Projektions-Pushdown durch. Alle Spalten werden aus Dateien gelesen, die den Prädikatfilter bestehen.

- Bei der Arbeit mit `custom.jdbc` Anschlüssen in AWS Glue hängt die Fähigkeit zum Drücken von Druck von der Quelle und dem Anschluss ab. Bitte überprüfen Sie die entsprechende Connector-Dokumentation, um zu überprüfen, ob und wie Pushdown in AWS Glue unterstützt wird.

Verwenden von Auto Scaling für AWS Glue

Auto Scaling ist für Ihre AWS Glue ETL, interaktive Sitzungen und Streaming-Jobs mit AWS Glue Version 3.0 oder höher verfügbar.

Das Aktivieren von Auto Scaling bietet die folgenden Vorteile:

- AWS Glue automatisches Hinzufügen und Entfernen von Workern aus dem Cluster abhängig von der Parallelität in jeder Phase oder jedemMikro-Stapel der Auftragsausführung.
- Dadurch müssen Sie nicht mehr experimentieren und entscheiden, wie viele Mitarbeiter Sie für Ihre AWS Glue ETL-Jobs einsetzen möchten.
- Bei der angegebenen maximalen Anzahl von Mitarbeitern AWS Glue wird die richtige Größe der Ressourcen für die Arbeitslast ausgewählt.
- Sie können sehen, wie sich die Größe des Clusters während der Auftragsausführung ändert, indem Sie sich die CloudWatch Metriken auf der Detailseite der Auftragsausführung in AWS Glue Studio ansehen.

Auto Scaling für AWS Glue ETL- und Streaming-Jobs ermöglicht das On-Demand-Scaling der Rechenressourcen Ihrer Jobs. AWS Glue Beim Hochskalieren auf Abruf können Sie nur die erforderlichen Rechenressourcen zunächst beim Start des Auftrags zuweisen und auch die erforderlichen Ressourcen je nach Bedarf während des Auftrags bereitstellen.

Auto Scaling unterstützt auch die dynamische Skalierung der AWS Glue Jobressourcen im Laufe eines Jobs. Wenn während einer Auftragsausführung mehr Ausführer von Ihrer Spark-Anwendung angefordert werden, werden dem Cluster mehr Worker hinzugefügt. Wenn der Ausführer ohne aktive Berechnungsaufgaben im Leerlauf verblieben ist, werden der Ausführer und der zugehörige Worker entfernt.

Zu den häufigsten Szenarien, in denen Auto Scaling bei den Kosten und der Auslastung Ihrer Spark-Anwendungen hilft, gehören:

- ein Spark-Treiber, der eine große Anzahl von Dateien in Amazon S3 auflistet oder einen Ladevorgang durchführt, während die Executoren inaktiv sind

- Spark-Stages laufen aufgrund von zu viel Provisioning mit nur wenigen Executoren
- Datenverzerrungen oder ungleichmäßiger Rechenbedarf in den Spark-Phasen

Voraussetzungen

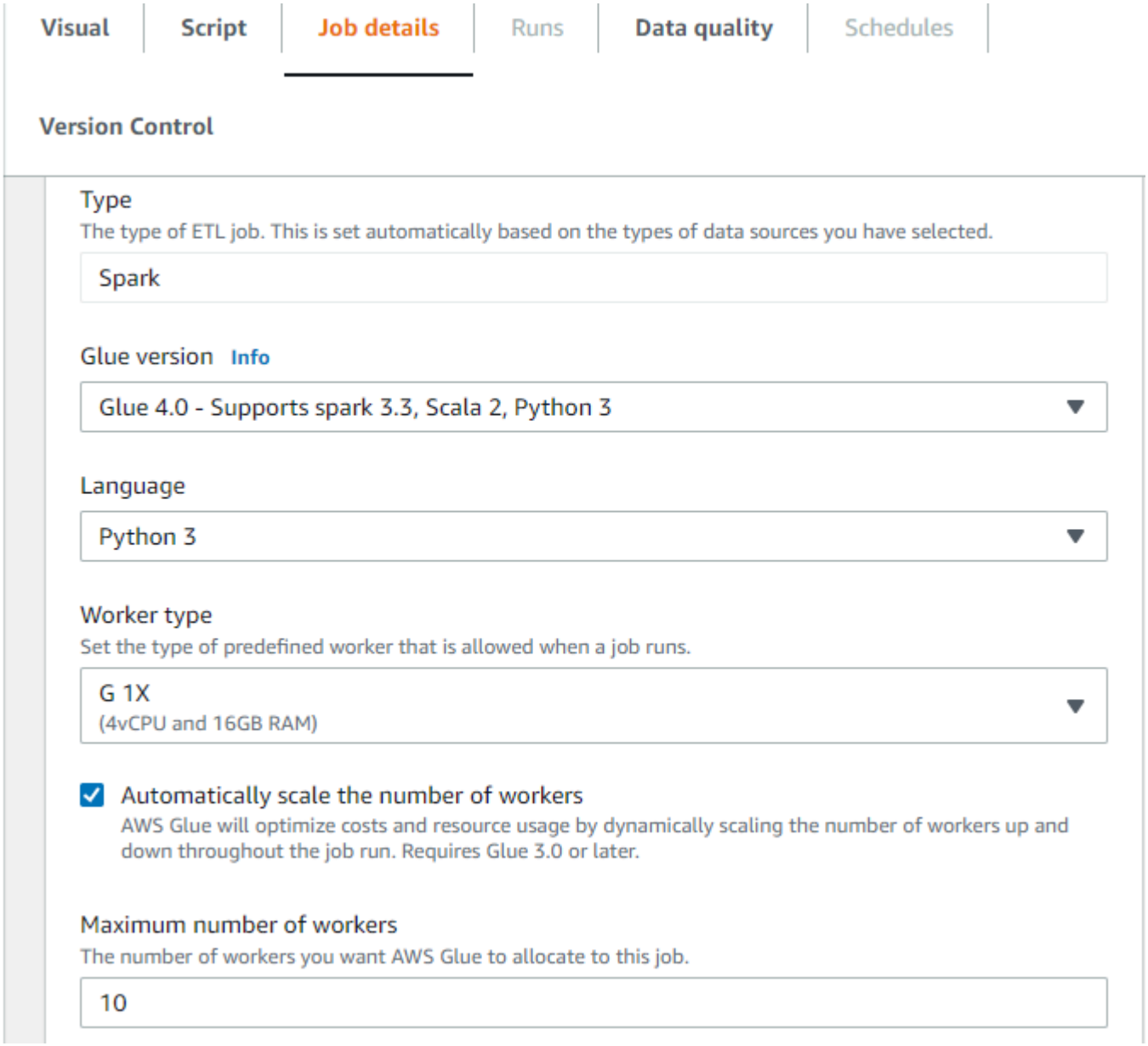
Auto Scaling ist nur für AWS Glue-Version 3.0 oder höher verfügbar. Um Auto Scaling zu verwenden, können Sie die Migrationsanleitung befolgen, um Ihre vorhandenen Aufträge auf AWS Glue-Version 3.0 oder höher zu migrieren oder neue Aufträge mit AWS Glue-Version 3.0 oder höher zu erstellen.

Auto Scaling ist für AWS Glue Jobs mit den Workertypen `G.1X` `G.2X` `G.4X` `G.8XG.12X`,`G.16X`,,`R.1X`,`R.2X`,`R.4X`,`R.8X`, oder `G.025X` (nur für Streaming-Jobs) verfügbar. Standard DPUs werden für Auto Scaling nicht unterstützt.

Aktivieren von Auto Scaling in AWS Glue Studio

Wählen Sie in AWS Glue Studio auf der Registerkarte Jobdetails den Typ als Spark oder Spark Streaming und die Glue-Version als **`Glue 3.0`** oder höher aus. Anschließend wird unter dem Worker-Typ ein Kontrollkästchen angezeigt.

- Wählen Sie die Option Automatisches Skalieren der Worker-Anzahl aus.
- Legen Sie die Maximale Worker-Anzahl fest, um die maximale Anzahl von Workern zu definieren, die für die Auftragsausführung ausgegeben werden können.



Auto Scaling mit der AWS CLI oder dem SDK aktivieren

Um Auto Scaling über die AWS CLI für Ihren Joblauf zu aktivieren, führen Sie `start-job-run` ihn mit der folgenden Konfiguration aus:

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
```

```
    "WorkerType": "G.2X", // G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X,
 and R.8X are supported for Auto Scaling Jobs
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Wenn die ETL-Auftragsausführung abgeschlossen ist, können Sie auch `get-job-run` aufrufen, um die tatsächliche Ressourcennutzung der Ausführung in DPU-Sekunden zu prüfen. Hinweis: Das neue Feld DPUSecondswird nur für Ihre Batch-Jobs auf AWS Glue Version 4.0 oder höher angezeigt, die mit Auto Scaling aktiviert sind. Dieses Feld wird für Streaming-Aufträge nicht unterstützt.

```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Sie können Auftragsausführungen mit Auto Scaling auch über das AWS Glue -SDK konfigurieren. Die Konfiguration ist dieselbe.

Auto Scaling mit interaktiven Sitzungen aktivieren

Informationen zur Aktivierung von Auto Scaling beim Erstellen von AWS Glue Jobs mit interaktiven Sitzungen finden Sie unter AWS Glue Interaktive Sitzungen konfigurieren.

Tipps und Überlegungen

Tipps und Überlegungen zur Feinabstimmung von AWS Glue Auto Scaling:

- Falls Sie keine Vorstellung vom Anfangswert der maximalen Anzahl von Arbeitern haben, können Sie mit der groben Berechnung beginnen, die in Estimate AWS Glue DPU erklärt wird. Sie sollten für Daten mit sehr geringem Datenvolumen keinen extrem hohen Wert für die maximale Anzahl von Mitarbeitern festlegen.
- AWS Glue Auto Scaling konfiguriert `spark.sql.shuffle.partitions` und `spark.default.parallelism` basiert auf der maximalen Anzahl von DPU (berechnet mit der

maximalen Anzahl von Arbeitern und dem Workertyp), die für den Job konfiguriert sind. Falls Sie bei diesen Konfigurationen den festen Wert bevorzugen, können Sie diese Parameter mit den folgenden Jobparametern überschreiben:

- Schlüssel: `--conf`
- Value (Wert): `spark.sql.shuffle.partitions=200 --conf spark.default.parallelism=200`

- Bei Streaming-Jobs erfolgt standardmäßig AWS Glue keine auto Skalierung innerhalb von Mikrobatches und es sind mehrere Mikrobatches erforderlich, um die auto Skalierung zu initiieren. Falls Sie Auto Scaling innerhalb von Mikrobatches aktivieren möchten, geben Sie `--auto-scale-within-microbatch` bitte an. Weitere Informationen finden Sie unter Job-Parameter-Referenz.

Überwachung von Auto Scaling mit CloudWatch Amazon-Metriken

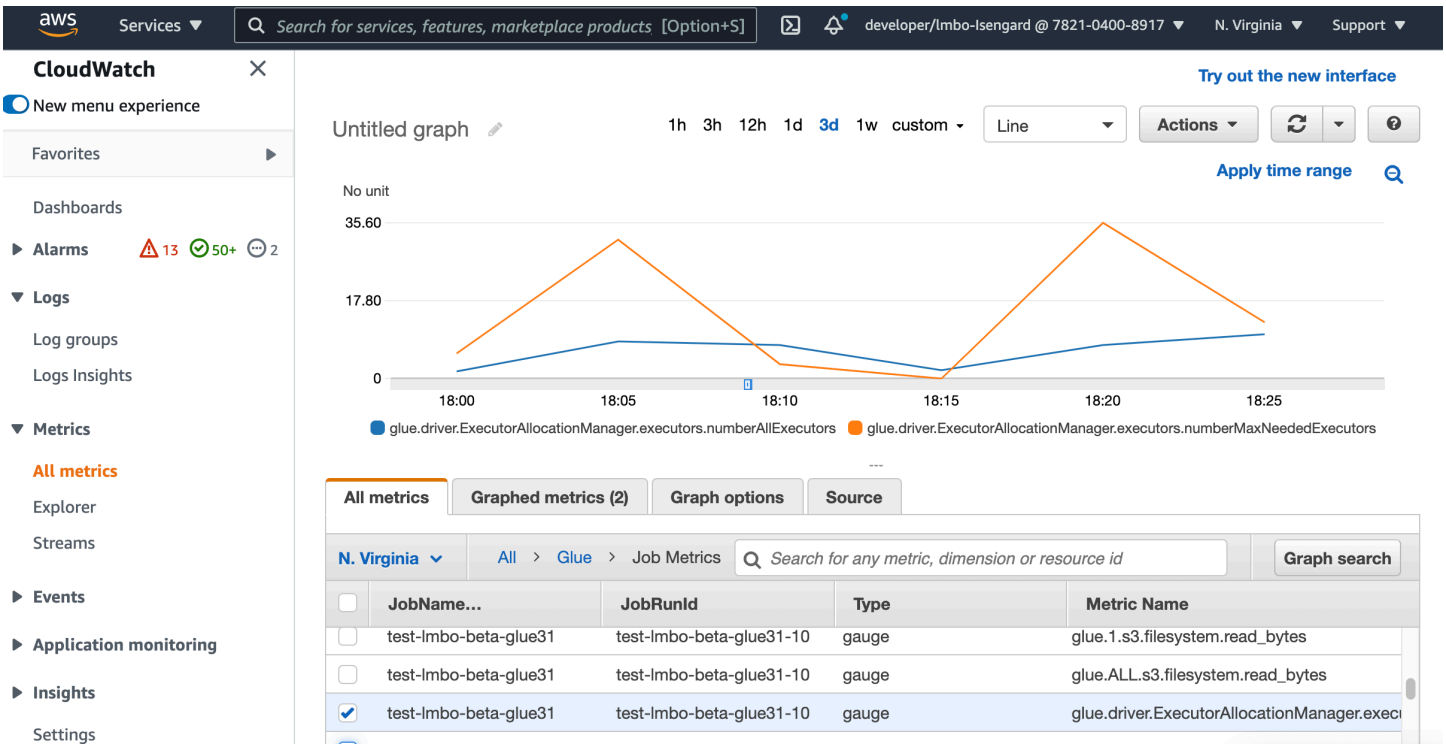
Die CloudWatch Executor-Metriken sind für Ihre Jobs der AWS Glue Version 3.0 oder höher verfügbar, wenn Sie Auto Scaling aktivieren. Die Metriken können verwendet werden, um die Nachfrage und die optimierte Nutzung von Ausführern in ihren Spark-Anwendungen, die mit Auto Scaling aktiviert sind, zu überwachen. Weitere Informationen finden Sie unter Überwachen AWS Glue Verwendung von CloudWatch Amazon-Metriken.

Sie können auch AWS Glue Observability-Metriken verwenden, um Einblicke in die Ressourcennutzung zu erhalten. Durch die Überwachung können Sie beispielsweise überwachen`glue.driver.workerUtilization`, wie viele Ressourcen mit und ohne Auto Scaling tatsächlich genutzt wurden. Ein anderes Beispiel: Durch die Überwachung von `glue.driver.skewness.job` und können Sie sehen`glue.driver.skewness.stage`, wie die Daten verzerrt sind. Diese Erkenntnisse helfen Ihnen bei der Entscheidung, Auto Scaling zu aktivieren und die Konfigurationen zu optimieren. Weitere Informationen finden Sie unter Überwachung mitÜberwachen mit AWS Glue Beobachtbarkeitsmetriken.

- glue.driver. ExecutorAllocationManager. Testamentsvollstrecker. numberAllExecutors
- Klebertreiber. ExecutorAllocationManager. Testamentsvollstrecker. numberMaxNeededTestamentsvollstrecker

Weitere Informationen zu diesen Metriken finden Sie unter Überwachung für die DPU-Kapazitätsplanung.

> **Note**
>
> CloudWatch Executor-Metriken sind für interaktive Sitzungen nicht verfügbar.

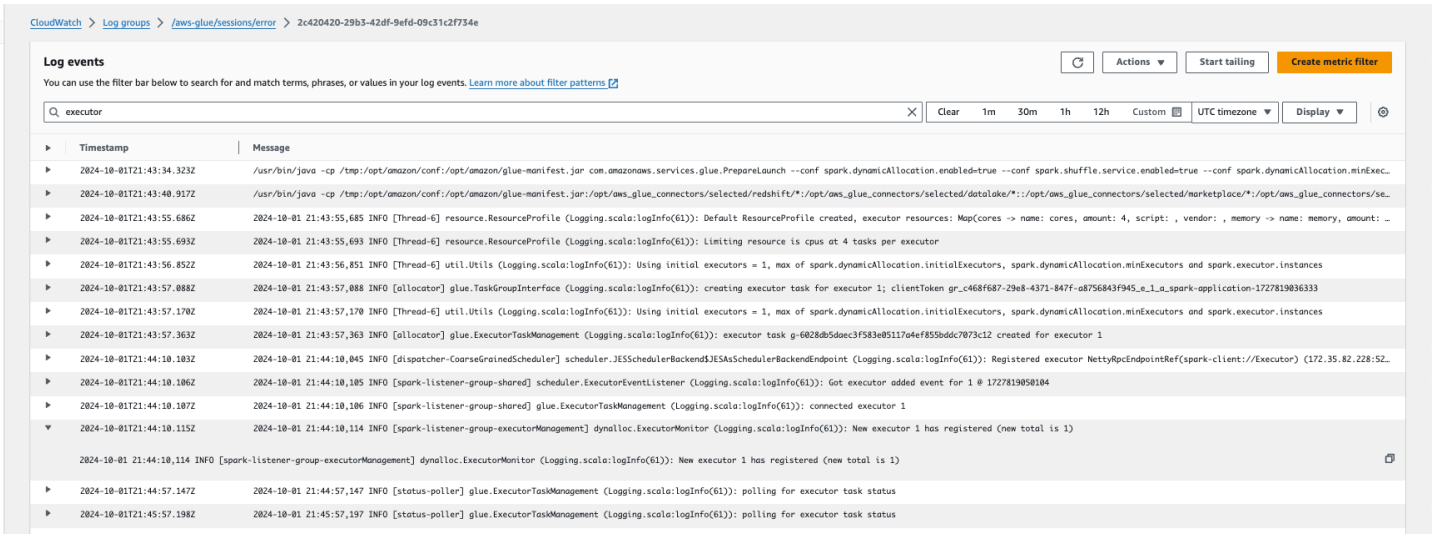


Überwachung von Auto Scaling mit Amazon CloudWatch Logs

Wenn Sie interaktive Sitzungen verwenden, können Sie die Anzahl der Executors überwachen, indem Sie kontinuierlich Amazon CloudWatch Logs aktivieren und in den Protokollen nach „Executor“ suchen, oder indem Sie die Spark-Benutzeroberfläche verwenden. Verwenden Sie dazu die `%%configure` Magie, um die kontinuierliche Protokollierung zusammen mit zu aktivieren. `enable auto scaling`

```
%%configure{
    "--enable-continuous-cloudwatch-log": "true",
    "--enable-auto-scaling": "true"
}
```

Suchen Sie in den Amazon CloudWatch Logs-Ereignissen in den Protokollen nach „Executor“:

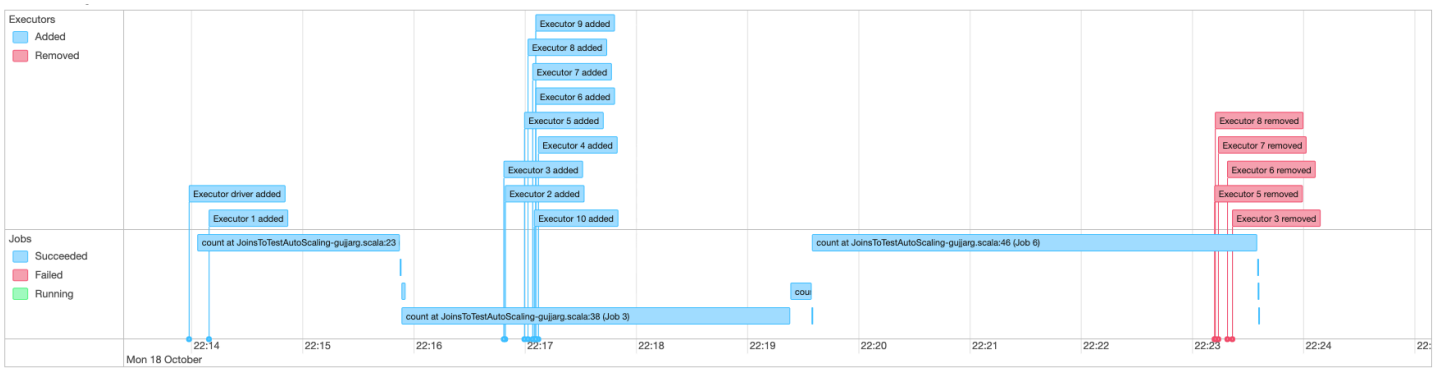


Überwachen von Auto Scaling mit der Spark-Benutzeroberfläche

Wenn Auto Scaling aktiviert ist, können Sie auch mit der Glue Spark-Benutzeroberfläche überwachen, dass Ausführer basierend auf der Nachfrage in Ihren AWS Glue-Aufträgen mit dynamischer Hoch- und Herabskalierung hinzugefügt und entfernt werden. Weitere Informationen finden Sie unter Aktivierung der Apache Spark-Weboberfläche für AWS Glue jobs.

Wenn Sie interaktive Sitzungen von Jupyter Notebook aus verwenden, können Sie die folgende Magie ausführen, um Auto Scaling zusammen mit der Spark-Benutzeroberfläche zu aktivieren:

```
%%configure{
    "--enable-auto-scaling": "true",
    "--enable-continuous-cloudwatch-log": "true"
}
```



Überwachen der DPU-Nutzung bei der Auto-Scaling-Auftragsausführung

Sie können die Ansicht AWS Glue Studio -Auftragsausführung verwenden, um die DPU-Nutzung Ihrer Auto-Scaling-Aufträge zu überprüfen.

1. Wählen Sie im Navigationsbereich die AWS Glue Studio Option Überwachung aus. Die Seite „Monitoring“ (Überwachung) wird angezeigt.
2. Scrollen Sie nach unten zur Tabelle „Job runs“ (Auftragsausführungen).
3. Navigieren Sie zur gewünschten Auftragsausführung und scrollen Sie zur Spalte „DPU hours“ (DPU-Stunden), um die Nutzungswerte für die entsprechende Ausführung zu prüfen.

Einschränkungen

AWS GlueStreaming Auto Scaling unterstützt derzeit keinen DataFrame Streaming-Join mit einer außerhalb von DataFrame erstellten Statik`ForEachBatch`. Eine innerhalb DataFrame von erstellte Statik `ForEachBatch` funktioniert erwartungsgemäß.

Workload-Partitionierung mit begrenzter Ausführung

Fehler in Spark-Anwendungen entstehen häufig durch ineffiziente Spark-Skripte, die verteilte Ausführung von großen Transformationen im Speicher und Datensatzanomalien. Es gibt viele Gründe, die beim Treiber oder Executor Probleme mit dem Arbeitsspeicher verursachen können, z. B. eine Datenverzerrung, eine Auflistung zu vieler Objekte oder eine große Datenmischung. Diese Probleme treten häufig auf, wenn Sie große Mengen an Backlog-Daten mit Spark verarbeiten.

AWS Glue ermöglicht es Ihnen, OOM-Probleme zu lösen und Ihre ETL-Verarbeitung durch Workload-Partitionierung zu vereinfachen. Bei aktivierter Workload-Partitionierung wählt jeder ETL-Auftragslauf nur unverarbeitete Daten aus, wobei eine Obergrenze für die Datensatzgröße oder die Anzahl der Dateien besteht, die mit diesem Auftragslauf verarbeitet werden können. Zukünftige Auftragsläufe verarbeiten die verbleibenden Daten. Wenn beispielsweise 1 000 Dateien verarbeitet werden müssen, können Sie die Anzahl der Dateien auf 500 festlegen und sie in zwei Auftragsläufe aufteilen.

Die Workload-Partitionierung wird nur für Amazon-S3-Datenquellen unterstützt.

Aktivieren der Workload-Partitionierung

Sie können die begrenzte Ausführung aktivieren, indem Sie die Optionen in Ihrem Skript manuell festlegen oder Katalogtabelleneigenschaften hinzufügen.

Die Workload-Partitionierung mit begrenzter Ausführung in Ihrem Skript aktivieren:

1. Um eine erneute Verarbeitung von Daten zu vermeiden, aktivieren Sie Auftragslesezeichen im neuen oder vorhandenen Auftrag. Weitere Informationen finden Sie unter Verfolgung verarbeiteter Daten anhand von Auftragslesezeichen.
2. Ändern Sie Ihr Skript und legen Sie das begrenzte Limit in den zusätzlichen Optionen im AWS Glue `getSource`API. Sie sollten auch den Transformationskontext für das Auftragslesezeichen festlegen, um das `state`-Element zu speichern. Zum Beispiel:

   Python

   ```
   glueContext.create_dynamic_frame.from_catalog(
       database = "database",
       table_name = "table_name",
       redshift_tmp_dir = "",
       transformation_ctx = "datasource0",
       additional_options = {
           "boundedFiles" : "500", # need to be string
         # "boundedSize" : "1000000000" unit is byte
       }
   )
   ```

   Scala

   ```
   val datasource0 = glueContext.getCatalogSource(
       database = "database", tableName = "table_name", redshiftTmpDir = "",
       transformationContext = "datasource0",
       additionalOptions = JsonOptions(
           Map("boundedFiles" -> "500") // need to be string
         //"boundedSize" -> "1000000000" unit is byte
       )
   ).getDynamicFrame()
   ```

   ```
   val connectionOptions = JsonOptions(
       Map("paths" -> List(baseLocation), "boundedFiles" -> "30")
   )
   val source = glueContext.getSource("s3", connectionOptions, "datasource0", "")
   ```

Die Workload-Partitionierung mit begrenzter Ausführung in Ihrer Tabelle für Data Catalog aktivieren:

1. Legen Sie die Schlüssel-Wert-Paare im `parameters`-Feld für Ihre Tabellenstruktur in Data Catalog fest. Weitere Informationen finden Sie unter Anzeigen und Bearbeiten von Tabellendetails.
2. Legen Sie die Obergrenze für die Datensatzgröße oder die Anzahl der verarbeiteten Dateien fest:

   - Legen Sie `boundedSize` auf die Zielgröße des Datensatzes in Bytes fest. Der Auftragslauf wird beendet, nachdem die Zielgröße aus der Tabelle erreicht wurde.
   - Legen Sie `boundedFiles` auf die Zielanzahl der Dateien fest. Der Auftragslauf wird beendet, nachdem die Zielanzahl der Dateien verarbeitet wurde.

> **Note**
>
> Sie sollten nur eine von `boundedSize` oder `boundedFiles` festlegen, da nur eine einzelne Begrenzung unterstützt wird.

Einrichtung einer AWS Glue Auslöser, um den Job automatisch auszuführen

Sobald Sie die begrenzte Ausführung aktiviert haben, können Sie eine einrichten AWS Glue Auslöser, um den Job automatisch auszuführen und die Daten in sequentiellen Läufen inkrementell zu laden. Gehe zum AWS Glue Konsole und erstellen Sie einen Trigger, richten Sie den Zeitplan ein und fügen Sie ihn Ihrem Job hinzu. Dann wird der nächste Auftragslauf automatisch ausgelöst und der neue Datenbatch verarbeitet.

Sie können auch Folgendes verwenden AWS Glue Workflows zur Orchestrierung mehrerer Jobs zur parallel Verarbeitung von Daten aus verschiedenen Partitionen. Weitere Informationen finden Sie unter AWS Glue Auslöser und AWS Glue Arbeitsabläufe.

Weitere Informationen zu Anwendungsfällen und Optionen finden Sie im Blog Optimieren von Spark-Anwendungen mit Workload-Partitionierung in AWS Glue.

Bekannte Probleme für AWS Glue

Beachten Sie die folgenden bekannten Probleme für AWS Glue.

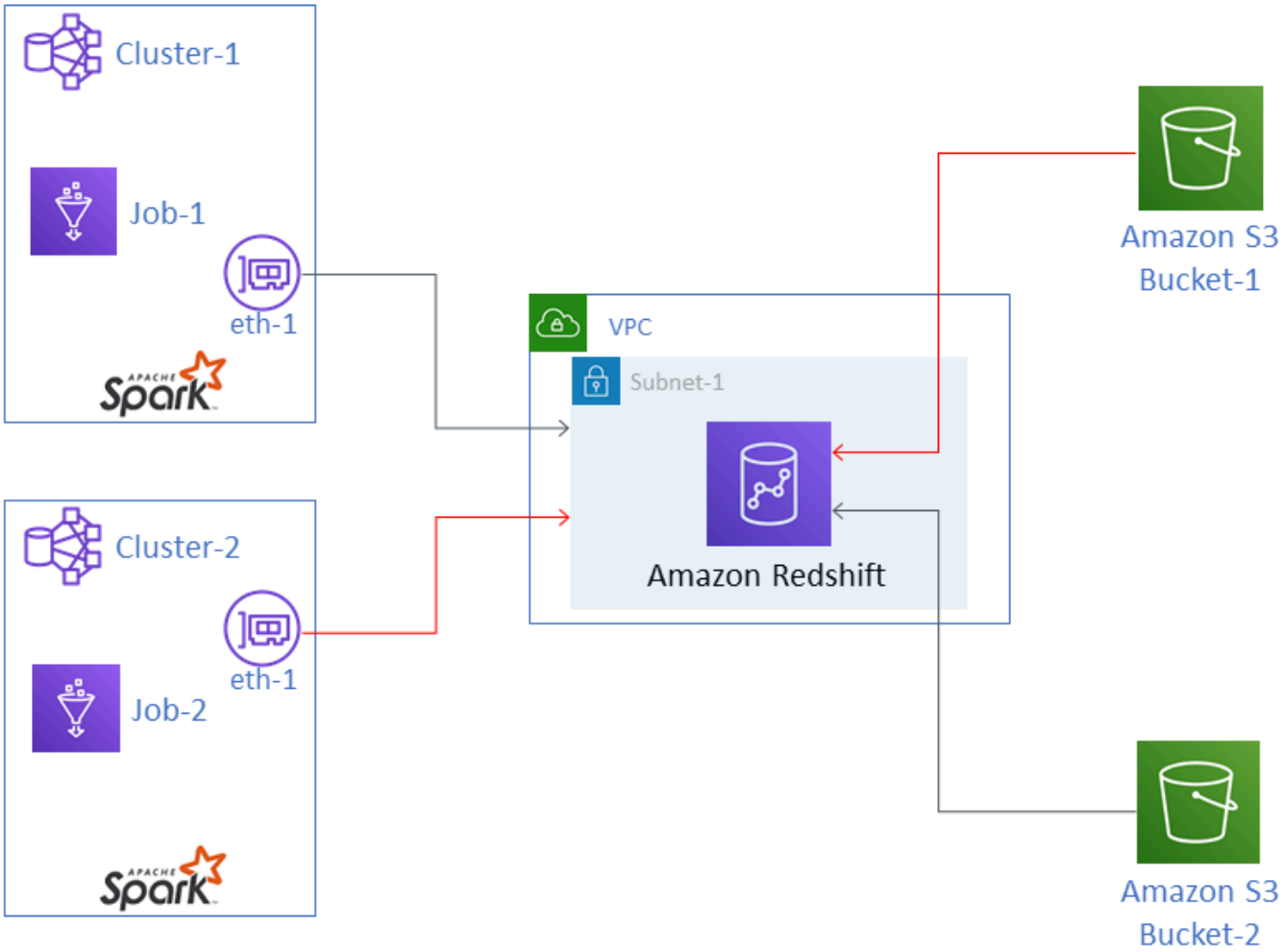
Themen

- Verhindern des auftragsübergreifenden Datenzugriffs

Verhindern des auftragsübergreifenden Datenzugriffs

Stellen Sie sich die Situation vor, in der Sie zwei haben AWS Glue Spark-Jobs in einem einzigen AWS Konto, von denen jeder in einem separaten Konto ausgeführt wird AWS Glue Spark-Cluster. Die Jobs verwenden AWS Glue Verbindungen für den Zugriff auf Ressourcen in derselben Virtual Private Cloud (VPC). In diesem Fall kann ein Auftrag, der in einem Cluster ausgeführt wird, auf die Daten des Auftrags zugreifen, der in dem anderen Cluster ausgeführt wird.

Das folgende Diagramm veranschaulicht ein Beispiel für diese Situation.



Im Diagramm AWS Glue `Job-1`läuft ein `Cluster-1` und Job-2 läuft ein. `Cluster-2` Beide Aufträge arbeiten mit derselben Instance von Amazon Redshift, die sich in `Subnet-1` einer VPC befindet. `Subnet-1` kann ein öffentliches oder privates Subnetz sein.

`Job-1`transformiert Daten aus Amazon Simple Storage Service (Amazon S3) `Bucket-1` und schreibt die Daten in Amazon Redshift. `Job-2`macht dasselbe mit den eingehenden Daten. `Bucket-2` `Job-1`verwendet die AWS Identity and Access Management (IAM-) Rolle `Role-1` (nicht gezeigt), die Zugriff auf `Bucket-1` gewährt. `Job-2`verwendet `Role-2` (nicht gezeigt), die Zugriff auf `Bucket-2` gewährt.

Diese Aufträge verfügen über Netzwerkpfade, die es ihnen ermöglichen, mit den Clustern des anderen zu kommunizieren und somit auf die Daten des anderen zuzugreifen. Beispielsweise könnte `Job-2` auf Daten in `Bucket-1` zugreifen. Im Diagramm wird dies als der Pfad in rot dargestellt.

Um diese Situation zu vermeiden, empfehlen wir, dass Sie verschiedene Sicherheitskonfigurationen an `Job-1` und `Job-2` anfügen. Durch das Anhängen der Sicherheitskonfigurationen wird der jobübergreifende Zugriff auf Daten durch Zertifikate blockiert, die AWS Glue erzeugt. Die Sicherheitskonfigurationen können Dummy-Konfigurationen sein. Das heißt, Sie können die Sicherheitskonfigurationen erstellen, ohne die Verschlüsselung von Amazon S3 S3-Daten, CloudWatch Amazon-Daten oder Job-Lesezeichen zu aktivieren. Alle drei Verschlüsselungsoptionen können deaktiviert werden.

Weitere Informationen zu den Sicherheitskonfigurationen finden Sie unter the section called “Verschlüsselung von Daten, die geschrieben wurden von AWS Glue”.

So fügen Sie eine Sicherheitskonfiguration an einen Auftrag an

1. Öffnen Sie die AWS Glue Konsole unter https://console.aws.amazon.com/glue/.
2. Erweitern Sie auf der Seite Configure the job properties (Auftragseigenschaften konfigurieren) für den Auftrag den Abschnitt Security configuration, script libraries and job parameters (Sicherheitskonfiguration, Skriptbibliotheken und Auftragsparameter).
3. Wählen Sie eine Sicherheitskonfiguration in der Liste aus.

Geschichte der Dokumentation für AWS Glue

Änderung	Beschreibung	Datum
Support für die neuen Worker-Typen G.12X, G.16X und speicheroptimierte R-Worker-Typen	Unterstützung für die neuen Workertypen G.12X, G.16X und die speicheroptimierte n Workertypen R.1X, R.2X, R.4X, R.8X für Jobs hinzugefü gt. AWS Glue Diese neuen Workertypen bieten zusätzlic he Rechen- und Speichero ptionen für unterschiedliche Workload-Anforderungen. Weitere Informationen finden Sie unter Jobs hinzufügen in AWS Glue.	30. Januar 2025
Support für 14 weitere neue native SaaS-Konnektoren für AWS Glue	Vierzehn weitere native SaaS-Konnektoren wurden hinzugefügt AWS Glue. Weitere Informationen finden Sie unter AWS Glue Verbindung hinzufügen.	30. Januar 2025
Support für 16 weitere neue native SaaS-Konnektoren für AWS Glue	Sechzehn weitere native SaaS-Konnektoren wurden hinzugefügt AWS Glue. Weitere Informationen finden Sie unter AWS Glue Verbindung hinzufügen.	17. Dezember 2024
Automatische Generierung von Spaltenstatistiken	AWS Glue Data Quality unterstützt jetzt Amazon SageMaker AI LakeHouse Tabellen und AWS Lake Formation verwaltete Iceberg-,	6. Dezember 2024

	Delta- und HUDI-Tabellen in Data Catalog und ETL. Weitere Informationen finden Sie unter AWS Glue Datenqualität.	
Support für Zero-ETL-Integrationen	Bei Zero-ETL handelt es sich um eine Reihe vollständig verwalteter Integrationen, mit denen der Bedarf an ETL-Datenpipelines minimiert wird. Weitere Informationen finden Sie unter Zero-ETL-Integrationen.	3. Dezember 2024
Support für wiederverwendbare Verbindungen	Ein neues AWS Glue Verbindungsschema bietet eine einheitliche Methode zur Verwaltung von Datenverbindungen zwischen AWS Diensten und Anwendungen wie AWS Glue Amazon Athena und Amazon SageMaker Unified Studio. Weitere Informationen finden Sie unter Verbindung zu Daten herstellen.	3. Dezember 2024

Support für AWS Glue Version 5.0.	Es wurden Informationen zur Unterstützung für AWS Glue Version 5.0 hinzugefügt. Zu den Funktionen gehören ein Apache Spark-Update auf 3.52, ein Java-Update auf 17, Aktualisierungen des Open-Table-Formats, eine native, fein abgestuft e Zugriffskontrolle von Sagemaker, Lakehouse- und Data Warehouse-Abstrakt ionsintegration, Unterstüt zung in Sagemaker Unified Studio und vieles mehr. Weitere Informationen finden Sie unter Versionshinweise und Migration von Jobs auf Version 5.0 AWS Glue . AWS GlueAWS Glue	3. Dezember 2024
Es wird eine Verbindung zum AWS Glue Iceberg-REST-Endpunkt AWS Glue Data Catalog hergestellt	AWS Glue Der Iceberg-R EST-Endpunkt unterstüt zt API-Operationen, die in der Apache Iceberg REST-Spezifikation spezifiziert sind. Mithilfe eines Iceberg-R EST-Clients können Sie Ihre Anwendung, die auf einer Analytics-Engine ausgeführ t wird, mit dem im Datenkata log gehosteten REST-Kata log verbinden. Weitere Informationen finden Sie unter Zugreifen auf den Datenkata log.	3. Dezember 2024

Automatische Generierung von Spaltenstatistiken	Generieren Sie automatisch Spaltenstatistiken für neue Tabellen in der AWS Glue Data Catalog. Weitere Informationen finden Sie unter Automatische Generierung von Spaltenstatistiken.	3. Dezember 2024
Support für generative KI-Upgrades für Apache Spark in AWS Glue	AWS Glue Mit Spark Upgrades in können Dateningenieure und Entwickler ihre vorhandenen AWS Glue Spark-Jobs mithilfe generativer KI aktualisieren und migrieren, aktualisieren und auf die neuesten Spark-Versionen migrieren. Weitere Informationen finden Sie unter Upgrade-Analyse mit KI.	22. November 2024
Support für generative KI-Fehlerbehebung für Apache Spark in AWS Glue	Generative KI-Fehlerbehebung für Apache Spark-Jobs in AWS Glue hilft Dateninge nieuren und Wissensch aftlern dabei, Probleme in ihren Spark-Anwendungen mühelos zu diagnostizieren und zu beheben. Weitere Informationen finden Sie unter Problembehandlung bei Spark-Jobs mit KI.	22. November 2024

Support für Iceberg-Optimierer für den Zugriff auf Amazon S3 S3-Buckets in einer VPC	AWS Glue Data Catalog unterstützt Iceberg-Tabellenoptimierer für den Zugriff auf Amazon S3 S3-Buckets von einer bestimmten Virtual Private Cloud (VPC) über eine Netzwerkverbindung. AWS Glue Weitere Informationen finden Sie unter Optimieren von Iceberg-Tabellen.	20. November 2024
Support für neun weitere neue native SaaS-Konnektoren für AWS Glue	Neun weitere native SaaS-Konnektoren wurden hinzugefügt AWS Glue. Weitere Informationen finden Sie unter AWS Glue Verbindung hinzufügen.	19. November 2024
Support für zehn neue native SaaS-Konnektoren für AWS Glue	Zehn native SaaS-Konnektoren wurden hinzugefügt AWS Glue. Weitere Informationen finden Sie unter AWS Glue Verbindung hinzufügen.	15. November 2024
Support für Job Run Queuing für AWS Glue ETL-Jobs	Sie können die Warteschlange für die Ausführung von Jobs aktivieren, um Jobs zu einem späteren Zeitpunkt auszuführen, wenn sie aufgrund von Servicequotas nicht sofort ausgeführt werden können. Weitere Informationen finden Sie unter Konfiguration der Auftragseigenschaften für Spark-Jobs in AWS Glue.	3. September 2024

Aktualisierte Richtlinienänderungen	Dokumentierte Änderungen an den AwsGlueSessionUserRestrictedNotebookServiceRoleRichtlinien AwsGlueSessionUserRestrictedNotebookPolicyund, die für die Unterstützung von Sitzungen mit dem tag-on-create Eigentümer-Tag-Schlüssel erforderlich sind. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	30. August 2024
Anomalieerkennung und dynamische Regeln sind jetzt allgemein verfügbar	AWS Glue Data Quality nutzt einen Algorithmus für maschinelles Lernen, um aus vergangenen Trends zu lernen und dann future Werte vorherzusagen, um Anomalien zu erkennen. Mit dynamischen Regeln können Sie dynamische Schwellenwerte angeben. Weitere Informationen finden Sie unter Optimieren der Abfrageleistung für Iceberg-Tabellen.	7. August 2024

Aktualisierte Richtlinienänderungen	Dokumentierte Änderungen an den AwsGlueSessionUserRestrictedServiceRoleRichtlinien AwsGlueSessionUserRestrictedPolicyund, die für die Unterstützung von Sitzungen mit dem tag-on-create Eigentümer-Tag-Schlüssel erforderlich sind. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	5. August 2024
Das Generieren von Spaltenstatistiken für Iceberg-Tabellen ist jetzt allgemein verfügbar	AWS Glue unterstützt die Berechnung und Aktualisierung der Anzahl unterschiedlicher Werte (NDVs) für jede Spalte in Eisberg-Tabellen. Weitere Informationen finden Sie unter Erkennung von Anomalien in den Bereichen AWS Glue Datenqualität und dynamische Regeln.	9. Juli 2024

Support für AWS Glue Nutzungsprofile	Administratoren können AWS Glue Nutzungsprofile für verschiedene Benutzerklassen innerhalb des Kontos erstellen, z. B. für Entwickler, Tester und Produktteams. Diese Flexibilität ermöglicht es Administratoren, für jede Benutzerklasse unterschiedliche Nutzungs- und Kostenkontrollen anzuwenden. Weitere Informationen finden Sie unter AWS Glue Nutzungsprofile einrichten.	18. Juni 2024
Support für einen Salesforce-Konnektor AWS Glue für Spark	Es wurden Informationen über einen neuen AWS Glue Connector für Salesforce hinzugefügt. Mit dieser Funktion können Sie Spark verwenden AWS Glue , um in Version AWS Glue 4.0 und späteren Versionen aus Salesforce zu lesen und in Salesforce zu schreiben. Weitere Informationen finden Sie unter Verbindung zu Salesforce herstellen.	22. Mai 2024

Amazon Q-Datenintegration in AWS Glue (GA)	Amazon Q Data Integration in AWS Glue ist eine neue generative KI-Funktion AWS Glue , die es Dateninge nieuren und ETL-Entwicklern ermöglicht, Datenintegrationsj obs in natürlicher Sprache zu erstellen. Ingenieure und Entwickler können Q bitten, Jobs zu verfassen , Probleme zu beheben AWS Glue und Fragen zur Datenintegration zu beantwort en. Weitere Informationen erhalten Sie unter Amazon-Q-Datenintegration in AWS Glue. Diese Funktion beinhalte t eine Aktualisierung der Richtlinien `AwsGlueSe ssionUserRestricte dPolicy` `AwsGlueSe ssionUserRestricte dNotebookServiceRo le` , und `AwsGlueSe ssionUserRestricte dServiceRole` AWS verwaltete Richtlinien. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlini en.	30. April 2024

Amazon Q-Datenintegration in AWS Glue (Vorschau)	Amazon Q Data Integration in AWS Glue ist eine neue generative KI-Funktion AWS Glue , die es Dateninge nieuren und ETL-Entwicklern ermöglicht, Datenintegrationsj obs in natürlicher Sprache zu erstellen. Ingenieure und Entwickler können Q bitten, Jobs zu verfassen, Probleme zu beheben AWS Glue und Fragen zur Datenintegration zu beantworten. Weitere Informationen erhalten Sie unter Amazon-Q-Dateninte gration in AWS Glue. Diese Funktion beinhaltet eine Aktualisierung der `AwsGlueSe ssionUserRestricte dNotebookPolicy` AWS verwalteten Richtlinie. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlini en.	30. Januar 2024

Aktualisierung der Dokumentation für AWS Glue Streaming	Es wurde ein neues Kapitel mit neuen und neu organisierten Inhalten für AWS Glue Streaming hinzugefügt. In diesem Inhalt wird beschrieben, wie Streaming funktioniert AWS Glue, welche Eigenschaften die Datenverarbeitung in Echtzeit hat und wie Sie Ihre Streaming-Jobs überwachen können. Weitere Informationen finden Sie unter AWS Glue - Streaming.	27. Dezember 2023
Unterstützung der detaillierten Erkennung von sensiblen Daten	Mit der Transformation „Detect Sensitive Data“ lassen sich Entitäten erkennen, maskieren oder entfernen, die Sie definieren oder die von AWS Glue vordefiniert werden. Mithilfe detaillierter Aktionen können Sie außerdem eine bestimmte Aktion pro Entität anwenden. Weitere Informationen finden Sie unter Verwenden der detaillierten Erkennung sensibler Daten.	26. November 2023

Support für die Überwachung von Jobs mit AWS Glue Observability-Metriken	Verwenden Sie AWS Glue Observability-Metriken, um Erkenntnisse darüber zu gewinnen, was in Ihren AWS Glue Apache Spark-Jobs passiert, und verbessern Sie so die Triaging und Analyse von Problemen. Weitere Informationen finden Sie unter Überwachung mit AWS Glue Observability-Metriken.	26. November 2023
Support für die Erkennung von Anomalien in der AWS Glue Datenqualität	AWS Glue Bei der Erkennung von Datenqualitätsanomalien werden Algorithmen für maschinelles Lernen (ML) auf Datenstatistiken im Laufe der Zeit angewendet, um abnormale Muster und versteckte Datenqualitätsprobleme zu erkennen, die durch Regeln schwer zu erkennen sind. Weitere Informationen finden Sie unter Erkennung von Anomalien im Bereich AWS Glue Datenqualität.	26. November 2023

Update auf das standardmäßige Protokollierungsverhalten der Spark-Benutzeroberfläche	Spark-Jobs, die Spark-UI-Logs generieren, schreiben jetzt mit einem anderen Dateinamenmuster, um die Spark-Benutzeroberfläche in der AWS Glue Konsole zu unterstützen. Dadurch wird das Verhalten der CloudWatch Protokolle nicht geändert. Sie können das alte Verhalten wiederherstellen, indem Sie die Auftragskonfiguration aktualisieren. Weitere Informationen finden Sie unter Überwachen von Aufträgen über die Web-UI von Apache Spark.	17. November 2023

Support für neue Datenquellen in AWS Glue Spark	Verbindungen zu Amazon OpenSearch Service, Azure SQL, Azure Cosmos for NoSQL, SAP HANA, Teradata Vantage und Vertica werden jetzt nativ unterstützt. AWS Glue Darüber hinaus sind Verbindungen zu diesen Datenquellen zusammen mit MongoDB jetzt für die Verwendung im Visual Editor von AWS Glue Studio verfügbar. Weitere Informationen zur Spark-Unterstützung finden Sie unter Verbindungstypen und Optionen AWS Glue für ETL in AWS Glue für Spark und unter AWS Glue Verbindung hinzufügen für Informationen zur Verwendung im visuellen Editor von AWS Glue Studio.	17. November 2023
Support für die Generierung von Spaltenstatistiken	Sie können Statistiken auf Spaltenebene für AWS Glue Data Catalog Tabellen in Datenformaten wie Parquet, ORC, JSON, ION, CSV und XML berechnen, ohne zusätzliche Daten-Pipelines einzurichten. Weitere Informationen finden Sie unter Arbeiten mit Spaltenstatistiken.	16. November 2023

Support für die Datenkomprimierung von Iceberg-Tabellen	Um die Leseleistung von AWS Analysediensten wie Amazon Athena und Amazon EMR sowie AWS Glue ETL-Jobs zu verbessern, bietet Data Catalog verwaltete Komprimierung (ein Prozess, der kleine Amazon S3 S3-Objekte zu größeren Objekten komprimiert) für Eisberg-T abellen im Datenkatalog. Weitere Informationen finden Sie unter Optimieren von Iceberg-Tabellen.	13. November 2023
Aktualisieren des Warteverhaltens bei der Auftragsausführung	Ausführungen von standardmäßigen Spark- und Python-Shell-Aufträgen werden jetzt in bestimmten Situationen zu `WAITING` wechseln, anstatt sofort zu `FAILED` zu wechseln. Weitere Informationen finden Sie unter Status von AWS Glue -Auftragsausführungen.	8. November 2023
AWS Glue Studio Das Benutzerhandbuch wurde im Entwicklerhandbuch zusammengefasst AWS Glue	Das AWS Glue Studio Benutzerhandbuch wurde in das Entwicklerhandbuch verschoben, um ein einheitliches Benutzerhandbuch für AWS Glue Studio die AWS Glue Konsole und den AWS Glue Studio programmatischen Zugriff zu schaffen.	25. Oktober 2023

Aktualisierung der AWSGlue ServiceNotebookRole AWS verwalteten Richtlinie	Es wurden Informationen über ein geringfügiges Update der AWSGlue ServiceNotebookRole AWS verwalteten Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	09. Oktober 2023
AWS Glue Studio unterstützt fünf neue integrierte Transformationen	AWS Glue Studio unterstützt die folgenden fünf neuen integrierten Transformationen: Datensatzabgleich, Nullzeilen entfernen, JSON-Spalte analysieren, JSON-Pfad extrahieren und Regex-Extraktor. Weitere Informationen finden Sie unter Transformationsknoten für verwaltete Daten bearbeiten AWS Glue.	11. August 2023
Aktualisierung der AWSGlue ServiceRole AWS verwalteten Richtlinie	Es wurden Informationen über ein geringfügiges Update der AWSGlue ServiceRole AWS verwalteten Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	4. August 2023

Unterstützung für das Crawling von Apache-Hudi-Tabellen	Es wurden Informationen zur Verwendung AWS Glue zum Crawlen von Hudi-Tabellen in Amazon S3 S3-Buckets und zur Registrierung der Hudi-Tabellen im hinzugefügt. AWS Glue Data Catalog Weitere Informationen finden Sie unter Welche Datenspeicher kann ich crawlen? und Crawler-Eigenschaften.	21. Juli 2023
Aktualisierung der verwalteten Richtlinie AWSGlueConsoleFullAccess AWS	Es wurden Informationen über ein geringfügiges Update der AWSGlue ConsoleFullAccess AWS verwalteten Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	14. Juli 2023
Unterstützung für das Crawling von Apache-Iceberg-Tabellen	Es wurden Informationen zur Verwendung AWS Glue zum Crawlen von Iceberg-Tabellen in Amazon S3 S3-Buckets und zur Registrierung der Iceberg-Tabellen im hinzugefügt. AWS Glue Data Catalog Weitere Informationen finden Sie unter Welche Datenspeicher kann ich crawlen? und Crawler-Eigenschaften.	07. Juli 2023

Support für AWS Glue mit Ray	Es wurden Informationen über AWS Glue with Ray hinzugefügt, eine neue Engine, die AWS Glue Jobs unterstützen kann. Bestehende Inhalte wurden AWS Glue mit Spark-Inhalten neu organisiert, um sie eindeutig zu verstehen.	30. Mai 2023
Support für AWS Glue Datenqualität (GA)	AWS Glue Data Quality ist jetzt allgemein verfügbar. AWS Glue Data Quality hilft Ihnen, die Qualität Ihrer Daten zu bewerten und zu überwachen. Informationen zur Verwendung von AWS Glue Data Quality mit Data Catalog finden Sie unter AWS Glue Datenqualität. Weitere Informationen zur AWS Glue Datenqualität für AWS Glue Studio finden Sie unter Evaluieren der Datenqualität mit AWS Glue Studio.	24. Mai 2023

Unterstützung für größere Worker-Typen für Apache Spark-Aufträge	Die Nutzung der `G.4X`- und `G.8X`-Worker-Typen für Apache Spark-Aufträge wird jetzt unterstützt. Diese Worker-Typen sind für Aufträge geeignet, deren Workloads Ihre anspruchsvollsten Transformationen, Aggregationen, Zusammenführungen und Abfragen enthalten. Weitere Informationen finden Sie unter Jobs hinzufügen in AWS Glue.	8. Mai 2023
Unterstützung für die Erstellung von Partitionsindizes beim Crawling von Tabellen	Es wurden Informationen darüber hinzugefügt, wie Crawler die Erstellung von Partitionsindizes für Tabellen unterstützen, die der Crawler erkennt. Weitere Informationen finden Sie unter Festlegen der Partitionsindex-Crawler-Konfigurationsoption.	24. April 2023
Unterstützung für Metriken zur Ressourcennutzung	Es wurden Informationen zur Anzeige der Ressourcennutzung des Dienstes und zur Konfiguration von Alarmen in Amazon hinzugefügt CloudWatch. Weitere Informationen finden Sie unter AWS Glue -Ressourcenüberwachung.	7. April 2023

Aktualisierung der AWSGlue ConsoleFullAccess AWS verwalteten Richtlinie	Es wurden Informationen über ein geringfügiges Update der AWSGlue ConsoleFullAccess AWS verwalteten Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	28. März 2023
Es wurde eine Anleitung zur Verwendung AWS Glue mit einem AWS SDK mit Beispielen hinzugefügt	Das AWS Glue Entwicklerhandbuch enthält zwei neue Abschnitte, die Informationen zur Verwendung AWS Glue mit einem AWS SDK enthalten. Weitere Informationen finden Sie unter AWS Glue Mit einem AWS SDK verwenden und unter Codebeispielen zur AWS Glue Verwendung AWS SDKs.	23. Februar 2023
Aktualisierung der Dokumentation für IAM mit AWS Glue	Informationen zur Verwendung von IAM mit wurden neu organisiert und hinzugefügt. AWS Glue Weitere Informationen finden Sie unter Identitäts- und Zugriffsverwaltung für AWS Glue.	15. Februar 2023

Support für die Ausführung von Streaming-ETL-Jobs in AWS Glue Version 4.0	Es wurden Informationen zur Unterstützung der Ausführung von Streaming-ETL-Aufträgen in Glue Version 4.0 und zu neuen Optionen für die Verbindung mit einem Kafka-Cluster oder einem Cluster von Amazon Managed Streaming für Apache Kafka und Amazon Kinesis Data Streams hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Streaming-ETL-Aufträgen in AWS Glue und Verbindungstypen und Optionen für ETL in AWS Glue.	8. Februar 2023
Unterstützung für das Crawling von MongoDB-Atlas-Datenquellen	Es wurden Informationen zur Verwendung AWS Glue zum Crawlen von MongoDB Atlas-Datenquellen hinzugefügt. Weitere Informationen finden Sie unter Welche Datenspeicher kann ich crawlen? , Verbindungseigenschaften von MongoDB und MongoDB Atlas und Verwenden einer MongoDB- oder MongoDB Atlas-Verbindung.	06. Februar 2023

Support für das Crawling von Delta-Lake-Tabellen mit einem nativen Delta-Lake-Konnektor	Es wurden Informationen zur Verwendung AWS Glue zum Crawlen von Delta Lake-Tabellen mithilfe eines nativen Delta Lake-Conn ectors hinzugefügt. Mit dieser Funktion können Sie AWS Abfrage-Engines verwenden , um das Delta-Transaktions protokoll direkt abzufragen und Funktionen wie Zeitreisen und ACID-Garantien zu nutzen und Ihre Delta Lake-Meta daten aus Amazon S3 S3-Transaktionsdateien mit dem Datenkatalog zu synchroni sieren, um Spaltenberechtigun gen für Ihre Abfragen in Lake Formation zu aktiviere n. Weitere Informationen finden Sie unter So geben Sie Konfigurationsoptionen für einen Delta Lake-Date nspeicher an und Abfragen von Delta-Lake-Tabellen.	15. Dezember 2022

Support für AWS Glue Datenqualität (Vorschau)	Support für AWS Glue Datenqualität (Vorschau) ist jetzt verfügbar. AWS Glue Data Quality hilft Ihnen, die Qualität Ihrer Daten zu bewerten und zu überwache n, wenn Sie AWS Glue 3.0 verwenden. Informationen zur Verwendung von AWS Glue Data Quality mit Data Catalog finden Sie unter AWS Glue Datenqualität (Vorschau). Weitere Informationen zur AWS Glue Datenqualität für AWS Glue Studio finden Sie unter Evaluieren der Datenqualität mit AWS Glue Studio.	30. November 2022
Support für einen neuen Amazon-Redshift-Spark-Konnektor mit neuen Features und Leistungsverbesserungen	Support ist jetzt verfügbar für einen neuen Amazon Redshift Spark-Connector mit einem neuen JDBC-Trei ber zur Verwendung mit AWS Glue ETL-Jobs zur Erstellung von Apache Spark-Anwendungen, die als Teil Ihrer Datenaufnahme- und Transformationspipelines Daten in Amazon Redshift lesen und darauf schreiben. Weitere Informationen finden Sie unter Verschieben von Daten zu und von Amazon Redshift.	29. November 2022

Support für AWS Glue Version 4.0.	Es wurden Informationen zur Unterstützung für AWS Glue Version 4.0 hinzugefügt. Zu den Features gehören native Unterstützung für Open-Data-Lake-Frameworks mit Apache Hudi, Delta Lake und Apache Iceberg sowie native Unterstützung für das Amazon-S3-basierte Cloud-Shuffle-Speicher-Plugin (ein Apache-Spark-Plugin) zur Verwendung von Amazon S3 für Shuffling und elastische Speicherkapazität. Weitere Informationen finden Sie unter AWS Glue Versionshinweise und Migration von AWS Glue Jobs auf AWS Glue Version 4.0.	28. November 2022
AWS Glue Studio bietet jetzt benutzerdefinierte visuelle Transformationen	Mit benutzerdefinierten visuellen Transformationen können Kunden geschäftsspezifische ETL-Logik definieren, wiederverwenden und in ihren Teams freigeben. Weitere Informationen finden Sie unter Benutzerdefinierte visuelle Transformationen .	28. November 2022

Support für die Verwendung des AWS Glue Crawlers zum Veröffentlichen von Metadaten für JDBC-Datenspeicher	Die Verwendung des AWS Glue Crawlers zum Veröffentlichen von Metadaten wie Kommentaren und Rohtypen im Datenkatalog für JDBC-Datenspeicher ist jetzt verfügbar. Weitere Informationen finden Sie unter Nach Crawler in Datenkatalogtabellen festgelegte Parameter, Crawler-Eigenschaften und Struktur. JdbcTarget	18. November 2022
Support für das Crawling von Snowflake-Datenspeichern	Support für das Crawlen von AWS Glue Snowflake-Tabellen und -Ansichten und für die Veröffentlichung der Metadaten im Datenkatalog als Tabelleneintrag ist jetzt verfügbar. Bei externen Snowflake-Tabellen in Amazon S3 durchsucht der Crawler auch den Amazon-S3-Speicherort und den Dateiformattyp der externen Tabelle und füllt sie als Tabellenparameter aus. Weitere Informationen finden Sie unter Welche Datenspeicher kann ich crawlen?, AWS Glue -Verbindungseigenschaften und Vom Crawler für Data-Catalog-Tabellen festgelegte Parameter.	18. November 2022

Support für eine verbesserte Shuffle-Verwaltung Ihrer Spark-Anwendungen	Support für ein neues Cloud-Shuffle-Speicher-Plugin für Apache Spark ist jetzt verfügbar. Weitere Informationen finden Sie unter AWS Glue -Spark-Shuffle-Plugin mit Amazon S3 und Cloud-Shuffle-Speicher-Plugin für Apache Spark.	15. November 2022
Unterstützung für Datenkatalogziele bei beschleunigten Crawls hinzugefügt, Amazon S3 S3-Ereignisbenachrichtigungen	Zusätzlich zur bestehenden Unterstützung für Amazon S3 S3-Ziele ist jetzt Unterstützung für beschleunigte Crawls für Datenkatalogziele mithilfe von Amazon S3 S3-Ereignisbenachrichtigungen verfügbar. Weitere Informationen finden Sie unter Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen.	13. Oktober 2022
Support für die Angabe der maximalen Anzahl von Tabellen, die ein Crawler erstellen kann	Ab sofort erhalten Sie Support für die Angabe der maximalen Anzahl von Tabellen, die der Crawler erstellen darf. Weitere Informationen erhalten Sie unter So geben Sie die maximale Anzahl von Tabellen an, die der Crawler erstellen darf.	6. September 2022

Support für Python 3.9 in Python-Shell-Jobs in AWS Glue	Support ist jetzt verfügbar für die Ausführung von Skripten, die mit Python 3.9 kompatibel sind AWS Glue, in Python-Shell-Jobs in und für die Auswahl der Verwendung vorgefert igter Bibliothekssätze. Weitere Informationen finden Sie unter Python-Shell-Aufträge in AWS Glue.	11. August 2022
Support für die Ausführung nicht dringender oder nicht zeitkritischer AWS Glue Aufträge auf freier Kapazität	Die Konfiguration flexibler Ausführungen von Aufträgen für nicht dringende Aufträge, wie z. B. Vorproduktionsauft räge, Tests und einmalige Datenübertragungen, wird jetzt unterstützt. Weitere Informati onen finden Sie unter Jobs hinzufügen in. AWS Glue	09. August 2022
Support für einen neuen Workertyp für Streaming-Aufträgen	Support für den Einsatz von `G.025X`-Worker-Typ für Streaming-Aufträgen mit geringem Volumen ist jetzt verfügbar. Weitere Informati onen finden Sie unter Jobs hinzufügen in AWS Glue.	14. Juli 2022
Support für die Verwendung von Kafka SASL in Verbindun gen AWS Glue	Support ist jetzt für die Verwendung von Kafka SASL in AWS Glue Verbindungen verfügbar. Weitere Informati onen finden Sie unter AWS Glue Kafka-Verbindungse igenschaften für die Client-Au thentifizierung.	5. Juli 2022

Unterstützung von Apache Kafka Connector für Protobuf-Schemas	Die Unterstützung von Apache Kafka Connector für Protobuf-Schemas ist jetzt verfügbar. Weitere Informationen finden Sie unter AWS Glue Schema Registry.	9. Juni 2022
Support für Auto Scaling for AWS Glue Jobs (GA)	Es wurden Informationen zur Verwendung von Auto Scaling für Jobs in AWS Glue Version 3.0 hinzugefügt, um Rechenressourcen dynamisch zu skalieren. Weitere Informationen finden Sie unter Auto Scaling für AWS Glue verwenden.	14. April 2022
Aktualisierung der Dokumentation zum AWS Glue Entwickeln und Testen von AWS Glue Jobskripten	Informationen zu den verfügbaren Entwicklungs- und Testmethoden für AWS Glue, einschließlich Anweisungen für die Entwicklung mit Docker, wurden neu organisiert und hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Jobskripte entwickeln und testen.	14. März 2022
Hinzufügen von Protokollpuffern (protobuf) als unterstütztes Datenformat für die Schemaregistrierung AWS Glue	Informationen über Protobuf als unterstütztes Datenformat (zusätzlich zu AVRO und JSON) hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Schema Registry.	25. Februar 2022

Support für das Crawling von Delta Lake-Tabellen	Es wurden Informationen zur Verwendung zum Crawlen von AWS Glue Delta Lake-Tabellen hinzugefügt. Weitere Informationen finden Sie unter So geben Sie Konfigurationsoptionen für einen Delta-Lake-Datenspeicher an.	24. Februar 2022
Support für AWS Glue berufliche Einblicke	Es wurden Informationen zur Verwendung von AWS Glue Job Insights hinzugefügt, um das Debuggen und die Optimierung von Jobs für Ihre AWS Glue Jobs zu vereinfachen. Weitere Informationen finden Sie unter Überwachung mit AWS Glue Job Insights.	8. Februar 2022
Support für das Crawling von Amazon-S3-gestützten Datenkatalog-Tabellen mit einem VPC-Endpunkt	Zusätzlich zu Amazon-S3-Datenspeichern können Sie Ihre Amazon-S3-gestützten Datenkatalog-Tabellen so konfigurieren, dass sie nur von einer Amazon-Virtual-Private-Cloud-Umgebung (Amazon VPC) zu Sicherheits-, Prüfungs- oder Kontrollzwecken aufgerufen werden. Weitere Informationen finden Sie untern Crawling eines Amazon-S3-Datenspeichers oder Amazon-S3-gestützten Datenkatalogs mit einem VPC-Endpunkt.	3. Februar 2022

Support für von Lake Formation verwaltete Tabellen	Es wurden Informationen zur AWS Glue Unterstützung für von Lake Formation verwaltete Tabellen hinzugefügt, die ACID-Transaktionen, automatische Datenkomprimierung und Zeitreise abfragen unterstützen. Weitere Informationen finden Sie unter AWS Glue -API und im AWS Lake Formation -Entwicklerhandbuch.	30. November 2021
Neue AWS verwaltete Richtlinien für interaktive Sitzungen und Notizbücher hinzugefügt	Neue verwaltete Richtlinien für IAM sorgten für mehr Sicherheit bei der Verwendung AWS Glue mit interaktiven Sitzungen und Notizbüchern. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien für AWS Glue.	30. November 2021
Glue Schema Registry wird jetzt mit Streaming-Aufträgen unterstützt	Sie können Streaming-Aufträge erstellen, die auf Tabellen zugreifen, die Teil des Glue Schema Registry sind. Weitere Informationen finden Sie unter AWS Glue Schema Registry und Hinzufügen von Streaming-ETL-Aufträgen in AWS Glue.	15. November 2021

Support für neue Machine-Learning-Features	Es wurden Informationen über neue Features für die „Find matches“-Machine-Learning-Transformation hinzugefügt, einschließlich inkrementeller Übereinstimmung und Match-Scoring. Weitere Informationen finden Sie unter Inkrementelle Übereinstimmungen und Schätzen der Qualität von Übereinstimmungen mithilfe von Match-Konfidenzwerten.	31. Oktober 2021
(Private Vorschau) Support für AWS Glue Flex-Jobs	Es wurden Informationen zur Konfiguration von AWS Glue Spark-Jobs mit einer flexiblen Ausführungsklasse hinzugefügt, die sich für zeitunabhängige Jobs eignet, deren Start- und Abschlusszeiten variieren können. Weitere Informationen finden Sie unter Jobs hinzufügen in. AWS Glue	29. Oktober 2021
Unterstützung von beschleunigten Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen	Es wurden Informationen zur Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen hinzugefügt. Weitere Informationen finden Sie unter Beschleunigung von Crawls mithilfe von Amazon S3-Ereignisbenachrichtigungen.	15. Oktober 2021

Zusätzliche Sicherheitskonfigurationsoptionen im Zusammenhang mit der Zugriffskontrolle und VPCs	Es wurden Informationen zur Konfiguration neuer Zugriffsberechtigungen AWS Glue und zur Konfiguration von hinzugefügt. VPCs Weitere Informationen finden Sie unter AWS Tags in AWS Glue, Identitätsbasierte Richtlinien (IAM-Richtlinien), die Einstellungen mithilfe von Bedingungsschlüsseln oder Kontextschlüsseln steuern und Alle AWS Aufrufe für Ihre VPC konfigurieren.	13. Oktober 2021
Unterstützung für VPC-Endpunktrichtlinien	Es wurden Informationen zur Unterstützung von Virtual Private Cloud (VPC)-Endpunktrichtlinien in hinzugefügt AWS Glue. Weitere Informationen finden Sie unter AWS Glue und Schnittstellen-VPC-Endpunkte (AWS PrivateLink).	11. Oktober 2021
Glue Studio ist jetzt in China verfügbar	AWS Glue Studio ist jetzt in den Regionen China, Peking und Ningxia verfügbar.	11. Oktober 2021

AWS Glue Studio bietet Notizbuch-Authoring für die interaktive Bearbeitung von Jobs	Notebooks unterstützen Sie beim Schreiben und Ausführen von Code, bei der Visualisierung der Ergebnisse und beim Austausch von Erkenntnissen. In der Regel verwenden Datenwissenschaftler Notebooks für Experimente und Aufgaben der Datenexploration. Weitere Informationen finden Sie unter Verwenden von Notebooks.	1. Oktober 2021
Direkter Zugriff auf Streaming-Quellen jetzt verfügbar	Wenn Sie Ihrem ETL-Auftrag im visuellen Editor Datenquellen hinzufügen, können Sie Informationen für den Zugriff auf den Datenstrom angeben, anstatt eine Data Catalog-Datenbank und -Tabelle verwenden zu müssen.	30. September 2021
Die Richtlinie zur AWS Glue Versionsunterstützung wurde dokumentiert	Es wurden Informationen zur AWS Glue Versionsunterstützungsrichtlinie und zum Ende der Lebenszyklusphasen für bestimmte AWS Glue Versionen hinzugefügt. Weitere Informationen finden Sie unter Richtlinie zur AWS Glue -Versionsunterstützung.	24. September 2021

Benutzerdefinierte Konnektoren können jetzt mit Datenvorschauen verwendet werden	Wenn Sie den Datenquellenknoten mit einem benutzerdefinierten Konnektor bearbeiten, können Sie eine Vorschau des Datasets anzeigen, indem Sie die Registerkarte Dat-Vorschau wählen. Weitere Informationen finden Sie unter Benutzerdefinierte Konnektoren	24. September 2021
Support für AWS Glue interaktive Sitzungen (private Vorschau)	(Private Vorschau) Es wurden Informationen zur Verwendung AWS Glue interaktiver Sitzungen hinzugefügt, um Spark-Workloads in der Cloud von einem beliebigen Jupyter-Notebook aus auszuführen. Interaktive Sitzungen sind die bevorzugte Methode für die Entwicklung Ihres ETL-Codes (AWS Glue Extrahieren, Transformieren und Laden), wenn Sie 2.0 oder höher verwenden AWS Glue . Weitere Informationen finden Sie unter AWS Glue Interaktive Sitzungen für Jupyter Notebook einrichten und ausführen.	24. August 2021

Unterstützung für das Erstellen von Workflows aus Vorlagen (GA)	Es wurden Informationen zum Programmieren allgemeiner Extract, Transform, Load (ETL)-Anwendungsfälle in Blueprints und dann zum Erstellen von Workflows aus Blueprints hinzugefügt. Ermöglicht Datenanalysten ein einfaches Erstellen und Ausführen komplexer ETL-Prozesse. Weitere Informationen finden Sie unter Ausführen von komplexen ETL-Aktivitäten mithilfe von Blueprints und Workflows in AWS Glue.	23. August 2021

Support für AWS Glue Version 3.0.	Es wurden Informationen zur Unterstützung von AWS Glue Version 3.0 hinzugefügt, die das Apache Spark 3.0-Engine-Upgrade zur Ausführung von Apache Spark-ETL-Jobs sowie andere Optimierungen und Upgrades unterstützt. Weitere Informationen finden Sie unter AWS Glue Versionshinweise und Migration von AWS Glue Jobs auf AWS Glue Version 3.0. Zu den weiteren Funktionen dieser Version gehören der AWS Glue Shuffle-Manager, ein vektorisierter SIMD-CSV-Reader und Katalogpartitionsprädikate. Weitere Informationen finden Sie unter AWS Glue Spark Shuffle Manager mit Amazon S3, Formatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue und Serverseitiges Filtern mithilfe von Katalogpartitionsprädikaten.	18. August 2021
AWS GovCloud (US) Region	AWS Glue Studio ist jetzt verfügbar in AWS GovCloud (US) Region	18. August 2021

Python-Shell-Authoring verfügbar in AWS Glue Studio	Beim Erstellen eines neuen Auftrags können Sie nun einen Python-Shell-Auftrag erstellen. Weitere Informationen finden Sie unter Starten der Auftragserstellung und Bearbeiten von Python-Shell-Aufträgen in AWS Glue Studio.	13. August 2021
Support beim Starten eines Workflows mit einem EventBridge Amazon-Event	Es wurden Informationen darüber hinzugefügt, wie AWS Glue Sie in einer ereignisg esteuerten Architektur ein Event-Nutzer sein können. Weitere Informationen finden Sie unter Starten eines AWS Glue Workflows mit einem EventBridge Amazon-Ereignis und Anzeigen der EventBridge Ereignisse, die einen Workflow gestartet haben.	14. Juli 2021
Hinzufügen von JSON als unterstütztes Datenformat für die AWS Glue Schemareg istrierung	Informationen über JSON als unterstütztes Datenform at (zusätzlich zu AVRO) hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Schema Registry.	30. Juni 2021

Erstellen Sie AWS Glue Streaming-Jobs ohne Datenkatalogtabelle	Die `create_data_frame_from_options` -Python-Funktion oder `getSource` für Scala-Skripte unterstützen das Erstellen von Streaming-ETL-Aufträgen, die direkt auf die Datenströme anstatt auf eine Data-Catalog-Tabelle verweisen.	15. Juni 2021
AWS Glue Transformationen für maschinelles Lernen unterstützen jetzt Schlüssel AWS Key Management Service	Sie können eine Sicherheitskonfiguration oder einen AWS KMS Schlüssel angeben, wenn Sie AWS Glue Machine Learning Learning-Transformationen mit der Konsole, der CLI oder dem AWS Glue APIs konfigurieren. Weitere Informationen finden Sie unter Verwenden der Datenverschlüsselung mit Machine Learning-Transformationen und AWS Glue -Machine-Learning-API.	15. Juni 2021
Aktualisierung der AWSGlue ConsoleFullAccess AWS verwalteten Richtlinie	Es wurden Informationen über ein geringfügiges Update der AWSGlue ConsoleFullAccess AWS verwalteten Richtlinie hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Aktualisierungen der AWS verwalteten Richtlinien.	10. Juni 2021

Anzeigen des Datensatzes Ihres Auftrags beim Bearbeiten und Erstellen von Aufträgen	Sie können die neue Registerkarte Data preview (Datenvorschau) für einen Knoten in Ihrem Auftragsdiagramm verwenden, um einen Auszug der von diesem Knoten verwendeten Daten anzusehen. Weitere Informationen finden Sie unter Verwenden von Datenvorschauen im visuellen Auftragseditor.	7. Juni 2021
Unterstützung für das Festlegen eines Werts, der den Speicherort der Tabelle für die Crawler-Ausgabe angibt.	Es wurden Informationen zum Festlegen eines Wertes hinzugefügt, der bei der Konfiguration der Crawler-Ausgabe den Speicherort der Tabelle angibt.. Weitere Informationen finden Sie unter Den Tabellenspeicherort festlegen.	4. Juni 2021
Unterstützung für das Crawling von Probedateien in einem Datensatz beim Crawling eines Amazon-S3-Datenspeichers	Informationen zum Crawling von Probedateien beim Crawling von Amazon S3 wurden hinzugefügt. Weitere Informationen finden Sie unter Crawler-Eigenschaften.	10. Mai 2021

Support für den AWS Glue optimierten Parkettschreiber	Es wurden Informationen zur Verwendung des AWS Glue optimierten Parquet Writers hinzugefügt DynamicFrames , um Tabellen mit der `parquet` Klassifizierung zu erstellen oder zu aktualisieren. Weitere Informationen finden Sie unter Erstellen von Tabellen, Aktualisieren des Schemas und Hinzufügen neuer Partitionen im Datenkatalog aus AWS Glue ETL-Jobs und Formatierungsoptionen für ETL-Eingaben und -Ausgaben in AWS Glue.	4. Mai 2021
Unterstützung für Kafka-Client-Authentifizierungspasswörter	Es wurden Informationen darüber hinzugefügt, wie Streaming-ETL-Jobs die SSL-Client-Zertifikatsauthentifizierung mit Apache Kafka-Stream-Producern AWS Glue unterstützen. Sie können jetzt ein benutzerdefiniertes Zertifikat angeben und gleichzeitig eine AWS Glue Verbindung zu einem Apache Kafka-Cluster definieren, das bei der Authentifizierung verwendet AWS Glue wird. Weitere Informationen finden Sie unter AWS Glue -Verbindungseigenschaften und Verbindungs-API.	28. April 2021

Unterstützung für die Verarbeitung von Daten aus Amazon Kinesis Data Streams in einem anderen Konto bei Streaming-ETL-Aufträgen	Informationen zum Erstellen eines ETL-Streaming-Auftrags zum Konsumieren von Daten aus Amazon Kinesis Data Streams in einem anderen Konto hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Streaming-ETL-Jobs in. AWS Glue	30. März 2021
SQL-Transformation verfügbar	Sie können mit einem SQL-Transformationsknoten Ihre eigene Transformation in Form einer SQL-Abfrage schreiben . Weitere Informationen finden Sie unter Verwenden einer SQL-Abfrage zum Transformieren von Daten.	23. März 2021
Unterstützung für das Erstellen von Workflows aus Vorlagen (öffentliche Vorschau)	(Öffentliche Vorschau) Es wurden Informationen zum Programmieren allgemeiner Extract, Transform, Load (ETL)-Anwendungsfälle in Blueprints und dann zum Erstellen von Workflows aus Blueprints hinzugefügt. Ermöglicht Datenanalysten ein einfaches Erstellen und Ausführen komplexer ETL-Prozesse. Weitere Informationen finden Sie unter Ausführen von komplexen ETL-Aktivitäten mithilfe von Blueprints und Workflows in AWS Glue.	22. März 2021

Konnektoren können für Datenziele verwendet werden	Die Verwendung eines benutzerdefinierten AWS Marketplace Connectors oder Connectors für Ihr Datenziel wird jetzt unterstützt. Weitere Informationen finden Sie unter Erstellen von Aufträgen mit benutzerdefinierten Konnektoren.	15. März 2021
Support von Metriken zur Spaltenwichtigkeit für AWS Glue Machine-Learning-Transformationen	Es wurden Informationen zur Anzeige von Metriken zur Spaltenwichtigkeit bei der Arbeit mit Transformationen für AWS Glue maschinelles Lernen hinzugefügt. Weitere Informationen finden Sie unter Arbeiten mit Machine-Learning-Transformationen auf der AWS Glue Konsole.	5. Februar 2021
Jobplanung jetzt verfügbar in AWS Glue Studio	Sie können einen Zeitplan für Ihre Auftragsläufe in AWS Glue Studio definieren. Sie können die Konsole verwenden, um einen einfachen Zeitplan zu erstellen oder einen komplexeren Zeitplan mit der UNIX-ähnlichen cron-Syntax definieren. Weitere Informationen finden Sie unter Planen von Auftragsausführungen.	21. Dezember 2020

AWS Glue Custom Connectors veröffentlicht	AWS Glue Mit Custom Connectors können Sie Connectors in entdecken und abonnieren AWS Marketplace. Wir haben auch AWS Glue Spark-Laufzeitschnittstellen zum Anschließen von Konnektoren veröffentlicht, die für Apache Spark Datasource, Athena Federated Query und JDBC entwickelt wurden. APIs Weitere Informationen finden Sie unter Konnektoren und Verbindungen mit verwenden. AWS Glue Studio	21. Dezember 2020
Support für die Ausführung von Streaming-ETL-Jobs in AWS Glue Version 2.0	Es wurden Informationen zum Support für das Ausführen von Streaming-ETL-Aufträgen in Glue Version 2.0 hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Streaming-ETL-Jobs in AWS Glue.	18. Dezember 2020
Unterstützung für die Workload-Partitionierung mit begrenzter Ausführung	Informationen zum Aktivieren der Workload-Partitionierung, um die oberen Grenzen für die Datensatzgröße oder die Anzahl der Dateien zu konfigurieren, die bei ETL-Auftragsläufen verarbeitet werden. Weitere Informationen finden Sie unter Workload-Partitionierung mit begrenzter Ausführung.	23. November 2020

Unterstützung für die erweiterte Partitionsverwaltung	Es wurden Informationen zur Verwendung von new hinzugefügt APIs , um einen Partitionsindex zu to/from einer vorhandenen Tabelle hinzuzufügen oder zu löschen. Weitere Informationen finden Sie unter Arbeiten mit Indizes.	23. November 2020
Support für die AWS Glue Schemaregistrierung	Es wurden Informationen zur Verwendung der AWS Glue Schemaregistry zur zentralen Erkennung, Steuerung und Weiterentwicklung von Schemas hinzugefügt. Weitere Informationen finden Sie unter AWS Glue Schema Registry.	19. November 2020
Unterstützung für das Grok-Eingabeformat in Streaming-ETL-Aufträgen	Informationen zum Anwenden von Grok-Mustern auf Streaming-Quellen wie Protokolldateien wurden hinzugefügt. Weitere Informationen finden Sie unter Anwenden von Grok-Mustern auf Streaming-Quellen.	17. November 2020
Support für das Hinzufügen von Tags zu Workflows auf der AWS Glue Konsole	Informationen zum Hinzufügen von Tags beim Erstellen eines Workflows mit der AWS Glue -Konsole. Weitere Informationen finden Sie unter Erstellen und Ausarbeiten eines Workflows mithilfe der AWS Glue Konsole.	27. Oktober 2020

Unterstützung für inkrementelles Ausführen von Crawlern	Es wurden Informationen zur Unterstützung des inkrementellen Ausführens von Crawlern hinzugefügt, bei denen das Crawling nur für Amazon-S3-Ordner ausgeführt wird, die seit der letzten Ausführung hinzugefügt wurden. Weitere Informationen finden Sie unter inkrementelles Crawling.	21. Oktober 2020
Support für die Schemaerkennung für Streaming-ETL-Datenquellen. Support für Avro Streaming-ETL-Datenquellen und selbstverwaltetes Kafka	Streaming-Aufträge zum Extrahieren, Transformieren und Laden (ETL) AWS Glue können jetzt automatisch das Schema eingehender Datensätze erkennen und Schemaänderungen pro Datensatz verarbeiten. Selbstverwaltete Kafka-Datenquellen werden jetzt unterstützt. Streaming-ETL-Aufträge unterstützen jetzt in Datenquellen das Avro-Format. Weitere Informationen finden Sie unter Streaming-ETL in AWS Glue, Definieren von Jobeigenschaften für einen Streaming-ETL-Job und Hinweise und Einschränkungen für Avro-Streaming-Quellen.	7. Oktober 2020

Unterstützung für das Crawling von MongoDB- und DocumentDB-Datenquellen	Es wurden Informationen zur Unterstützung für das Crawling MongoDB- und Amazon-DocumentDB-Datenquellen (mit MongoDB-Kompatibilität) hinzugefügt. Weitere Informationen finden Sie unter Definieren von Crawlern.	5. Oktober 2020
Unterstützung für FIPS-Compliance	Es wurden Informationen zu FIPS-Endpunkten für Kunden hinzugefügt, die für den Zugriff auf Daten mit AWS Glue FIPS 140-2-validierte kryptographische Module benötigen. Weitere Informationen finden Sie unter FIPS-Compliance.	23. September 2020
AWS Glue Studio bietet eine benutzerfreundliche visuelle Oberfläche zum Erstellen und Überwachen von Jobs	Sie können jetzt eine einfache grafikbasierte Schnittstelle verwenden, um Aufträge zu erstellen, die Daten verschieben, transformieren und auf AWS Glue ausführen. Mit dem Dashboard zum Aufführen von Aufträgen in AWS Glue Studio können Sie die ETL-Ausführungen überwachen und dafür sorgen, dass die Aufträge wie vorgesehen laufen. Weitere Informationen finden Sie im AWS Glue Studio-Benutzerhandbuch.	23. September 2020

Unterstützung für das Erstellen von Tabellenindizes zur Verbesserung der Abfragele istung	Es wurden Informationen zum Erstellen von Tabellenindizes hinzugefügt, mit denen Sie eine Teilmenge der Partition en aus einer Tabelle abrufen können. Weitere Informationen finden Sie unter Arbeiten mit Indizes.	9. September 2020
Unterstützung für kürzere Startupzeiten bei der Ausführung von Apache-Sp ark-ETL-Aufträgen in AWS Glue Version 2.0	Es wurden Informationen zur Unterstützung für AWS Glue Version 2.0 hinzugefügt, die eine verbesserte Infrastruktur für die Ausführung von Apache Spark-ETL-Jobs mit kürzeren Startzeiten, Änderungen in der Protokollierung und Unterstützung für die Angabe zusätzlicher Python-Module auf Jobebene bietet. Weitere Informationen finden Sie unter AWS Glue -Versionshinweise und Ausführen von Spark-ETL-Aufträgen mit verkürzten Startupzeiten	10. August 2020
Unterstützung für die Begrenzung der Anzahl gleichzeitiger Workflow-Ausführungen	Es wurden Informationen zur Begrenzung der Anzahl gleichzeitiger Workflow-Ausführungen für einen bestimmten Workflow hinzugefügt. Weitere Informati onen finden Sie unter Erstellen und Ausarbeiten eines Workflows mithilfe der AWS Glue Konsole.	10. August 2020

Unterstützung für das Crawling eines Amazon-S3-Datenspeichers mit einem VPC-Endpunkt	Es wurden Informationen zur Konfiguration Ihres Amazon-S3-Datenspeichers für den Zugriff über eine Amazon Virtual Private Cloud (Amazon VPC)-Umgebung zu Sicherheits-, Überwachungs- oder Kontrollzwecken hinzugefügt. Weitere Informationen finden Sie untern Crawling eines Amazon-S3-Datenspeichers mit einem VPC-Endpunkt.	7. August 2020
Unterstützung für die Fortsetzung der Workflow-Ausführung	Es wurden Informationen zum Fortsetzen von Workflow-Ausführungen hinzugefügt, die nur teilweise abgeschlossen wurden, da ein oder mehrere Knoten (Aufträge oder Crawler) nicht erfolgreich abgeschlossen wurden. Weitere Informationen finden Sie unter Reparieren und Fortsetzen einer Workflow-Ausführung.	27. Juli 2020

Unterstützung für das Aktivieren privater CA-Zertifikate in Kafka-Verbindungen in AWS Glue.	Es wurden Informationen zu neuen Verbindungsoptionen hinzugefügt, die das Aktivieren privater CA-Zertifikate für Kafka-Verbindungen in AWS Glue unterstützen. Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue und Spezielle Parameter, die von verwendet werden AWS Glue.	20. Juli 2020
Unterstützung für das Lesen von DynamoDB-Daten in einem anderen Konto	Es wurden Informationen zur AWS Glue Unterstützung für das Lesen von Daten aus der DynamoDB-Tabelle eines anderen AWS Kontos hinzugefügt. Weitere Informationen finden Sie unter Aus DynamoDB-Daten in einem anderen Konto lesen.	17. Juli 2020
Support für eine DynamoDB-Writer-Verbindung in AWS Glue Version 1.0 oder höher	Informationen zur Unterstützung von DynamoDB-Writer sowie neue oder aktualisierte Verbindungsoptionen für DynamoDB zum Lesen oder Schreiben hinzugefügt. Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue.	17. Juli 2020

Unterstützung für Ressourcen-Links und kontoübergreifende Zugriffskontrolle mithilfe von AWS Glue und Lake Formation	Es wurden Inhalte zu neuen Datenkatalogobjekten hinzugefügt, die als Ressourcenlinks bezeichnet werden, und darüber, wie die gemeinsame Nutzung von Datenkatalogressourcen zwischen Konten sowohl mit als auch AWS Glue verwaltet werden kann. AWS Lake Formation Weitere Informationen finden Sie unter Gewährung von kontenübergreifendem Zugriff und Ressourcen-Links zu Tabellen.	7. Juli 2020
Unterstützung für das beispielhafte Abfragen von Datensätzen beim Crawling von DynamoDB-Datenspeichern	Es wurden Informationen zu neuen Eigenschaften hinzugefügt, die Sie beim Crawling eines DynamoDB-Datenspeichers konfigurieren können. Weitere Informationen finden Sie unter Crawler-Eigenschaften.	12. Juni 2020
Unterstützung für das Anhalten einer Workflow-Ausführung	Es wurden Informationen zum Beenden einer Workflow-Ausführung für einen bestimmten Workflow hinzugefügt. Weitere Informationen finden Sie unter Anhalten einer Workflow-Ausführung.	14. Mai 2020

Unterstützung für Spark-Streaming-ETL-Aufträge	Es wurden Informationen zum Erstellen von ETL-Aufträgen (Extrahieren, Transformieren und Laden) mit Streaming-Datenquellen hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Streaming-ETL-Aufträgen in AWS Glue.	27. April 2020
Unterstützung für Erstellen von Tabellen, Aktualisieren des Schemas und Hinzufügen neuer Partitionen zum Data Catalog nach Ausführung eines ETL-Auftrags	Es wurden Informationen hinzugefügt, die das Erstellen von Tabellen, das Aktualisieren des Schemas und das Hinzufügen neuer Partitionen erläutern, damit die Ergebnisse des ETL-Auftrags im Data Catalog angezeigt werden. Weitere Informationen finden Sie unter Erstellen von Tabellen, Aktualisieren des Schemas und Hinzufügen neuer Partitionen im Datenkatalog aus AWS Glue ETL-Jobs.	2. April 2020

Support für die Angabe einer Version für das Apache Avro-Datenformat als ETL-Eingabe und -Ausgabe in AWS Glue	Es wurden Informationen zum Angeben einer Version für das Apache Avro-Date nformat als ETL-Eingabe und -Ausgabe in AWS Glue hinzugefügt. Die Standardv ersion: 1.7. Mit der Formatopt ion `version` können Sie Avro Version 1.8 angeben, um das Lesen/Schreiben logischer Typen zu aktivieren. Weitere Informationen finden Sie unter Formatoptionen für ETL-Eingaben und -Ausgaben in AWS Glue.	31. März 2020
Unterstützung für den S3-optimierten EMRFS-Committer zum Schreiben von Parquet-Daten in Amazon S3	Es wurden Informationen zum Setzen eines neuen Flags hinzugefügt, um den S3-optimierten EMRFR-Committer für das Schreiben von Parquet-Daten nach Amazon S3 beim Erstellen oder Aktualisieren eines AWS Glue -Auftrags zu aktivieren. Weitere Informationen finden Sie unter Spezielle Parameter, die von verwendet werden AWS Glue.	30. März 2020

Die Support für maschinelles Lernen wird zu einer Ressource, die durch AWS Ressourcen-Tags verwaltet wird	Es wurden Informationen zur Verwendung von AWS Ressourcen-Tags zur Verwaltung und Steuerung des Zugriffs auf Ihre maschinellen Lerntransformationen in hinzugefügt. AWS Glue Sie können Jobs, Triggern, Endpunkten, Crawlern und Transformationen für maschinelles Lernen AWS Ressourcen-Tags zuweisen. AWS GlueWeitere Informationen finden Sie unter Tags AWS in. AWS Glue	2. März 2020
Unterstützung für nicht überschreibbare Auftragsargumente	Es wurden Informationen zur Unterstützung spezieller Aufgabenparameter hinzugefügt, die weder in Auslösern noch beim Ausführen der Aufgabe überschrieben werden können. Weitere Informationen finden Sie unter Hinzufügen von Aufträgen in AWS Glue.	12. Februar 2020

Unterstützung für neue Transformationen für die Arbeit mit Datensätzen in Amazon S3	Es wurden Informationen zu neuen Transformationen (Merge, Purge und Transitio n) und Amazon-S3-Speicher klassenausschlüssen hinzugefügt, damit Apache-Spark-Anwendungen mit Datensätzen in Amazon S3 arbeiten können. Weitere Informationen zur Unterstüt zung dieser Transformationen für Python finden Sie unter mergeDynamicFrameund Working with Datasets in Amazon S3. Informationen zu Scala finden Sie unter mergeDynamicFramesund Scala.AWS Glue GlueContext APIs	16. Januar 2020
Unterstützung für das Aktualisi eren von Data Catalog mit neuen Partitionsinformationen aus einem ETL-Auftrag	Es wurden Informationen zum Codieren eines ETL-Skrip ts (Extrahieren, Transform ieren und Laden) hinzugefü gt, um es AWS Glue Data Catalog mit neuen Partition sinformationen zu aktualisi eren. Mit diesem Feature müssen Sie den Crawler nach Abschluss des Auftrags nicht mehr erneut ausführen, um die neuen Partitionen anzuzeige n. Weitere Informationen finden Sie unter Aktualisieren von Data Catalog mit neuen Partitionen.	15. Januar 2020

Neues Tutorial: Verwenden eines SageMaker KI-Notebooks	Es wurde ein Tutorial hinzugefügt, das zeigt, wie Sie ein SageMaker Amazon-Notizbuch verwenden können, um Ihre ETL- und Machine-Learning-Skripte zu entwickeln. Siehe Tutorial: Verwenden Sie ein SageMaker Amazon-Notebook mit Ihrem Entwicklungsendpunkt.	3. Januar 2020
Unterstützung für das Lesen aus MongoDB und Amazon DocumentDB (mit MongoDB-Kompatibilität)	Es wurden Informationen über neue Verbindungstypen und Verbindungsoptionen zum Lesen aus und Schreiben in MongoDB und Amazon DocumentDB (mit MongoDB-Kompatibilität) hinzugefügt. Weitere Informationen finden Sie unter Verbindungstypen und Optionen für ETL in AWS Glue.	17. Dezember 2019

Verschiedene Korrekturen und Klärungen	Überall wurden Korrekturen und Klärungen hinzugefügt. Einträge aus dem Kapitel „Bekannte Probleme“ wurden entfernt. Es wurden Warnungen hinzugefügt, die nur symmetrische Kundenhauptschlüssel (CMKs) AWS Glue unterstützen, wenn Verschlüsselungseinstellungen für den Datenkatalog angegeben und Sicherheitskonfigurationen erstellt werden. Es wurde ein Hinweis hinzugefügt, der das Schreiben in Amazon DynamoDB AWS Glue nicht unterstützt.	9. 2019. Dezember 2019
Unterstützung für benutzerdefinierte JDBC-Treiber	Es wurden Informationen zur Verbindung mit Datenquellen und Zielen mit JDBC-Treibern hinzugefügt, die AWS Glue nicht nativ unterstützt werden, wie MySQL Version 8 und Oracle Database Version 18. Weitere Informationen finden Sie unter JDBC-connectionType-Werte.	25. November 2019

Support für die Verbindung von SageMaker KI-Notebooks mit verschiedenen Entwicklungsendpunkten	Es wurden Informationen darüber hinzugefügt, wie Sie ein SageMaker KI-Notebook mit verschiedenen Entwicklungsendpunkten verbinden können. Aktualisierungen zur Beschreibung der neuen Konsolenaktion für den Wechsel zu einem neuen Entwicklungsendpunkt und der neuen SageMaker KI-IAM-Richtlinie. Weitere Informationen finden Sie unter Arbeiten mit Notebooks auf der AWS Glue Konsole und Erstellen einer IAM-Richtlinie für Amazon SageMaker AI-Notebooks.	21. November 2019
Support für AWS Glue Versionen in Transformationen für maschinelles Lernen	Es wurden Informationen zur Definition der AWS Glue Version in einer Transformation für maschinelles Lernen hinzugefügt, um anzugeben, mit welcher Version AWS Glue einer Transformation für maschinelles Lernen kompatibel ist. Weitere Informationen finden Sie unter Arbeiten mit Machine-Learning-Transformationen auf der AWS Glue Konsole.	21. November 2019

Unterstützung für das Zurückspulen Ihrer Auftragslesezeichen	Es wurden Informationen zum Zurückspulen Ihrer Auftragslesezeichen zu jeder beliebigen vorherigen Auftragsausführung hinzugefügt, was dazu führt, dass die nachfolgende Auftragsausführung nur Daten aus der mit dem Lesezeichen versehenen Auftragsausführung neu verarbeitet. Beschrieben werden zwei neue Unteroptionen für die `job-bookmark-pause` -Option, mit denen Sie einen Auftrag zwischen zwei Lesezeichen ausführen können. Weitere Informationen finden Sie unter Verfolgen verarbeiteter Daten mithilfe von Job-Lesezeichen und speziellen Parametern, die von verwendet werden AWS Glue.	22. Oktober 2019
Unterstützung für benutzerdefinierte JDBC-Zertifikate für die Verbindung mit einem Datenspeicher	Es wurden Informationen zur AWS Glue Unterstützung von benutzerdefinierten JDBC-Zertifikaten für SSL-Verbindungen zu AWS Glue Datenquellen oder Zielen hinzugefügt. Weitere Informationen finden Sie unter Arbeiten mit Verbindungen in der AWS Glue -Konsole.	10. Oktober 2019

Unterstützung für Python Wheel	Es wurden Informationen zur AWS Glue Unterstützung von Wheel-Dateien (zusammen mit EGG-Dateien) als Abhängigkeiten für Python-Shell-Jobs hinzugefügt. Weitere Informationen finden Sie unter Bereitstellen Ihrer eigenen Python-Bibliothek.	26. September 2019
Support für die Versionierung von Entwicklungsendpunkten in AWS Glue	Es wurden Informationen zur Definition der `Glue version` in der Entwicklung befindlichen Endpunkte hinzugefügt. `Glue version`bestimmt die Versionen von Apache Spark und Python, die AWS Glue unterstützt werden. Weitere Informationen finden Sie unter Hinzufügen eines Entwicklungsendpunkts.	19. September 2019
Support für die Überwachung AWS Glue mithilfe der Spark-Benutzeroberfläche	Es wurden Informationen zur Verwendung der Apache Spark-Benutzeroberfläche zum Überwachen und Debuggen von AWS Glue ETL-Jobs, die auf dem AWS Glue Jobsystem ausgeführt werden, und von Spark-Anwendungen auf AWS Glue Entwicklungsendpunkten hinzugefügt. Weitere Informationen finden Sie unter Überwachung AWS Glue mithilfe der Spark-Benutzeroberfläche.	19. September 2019

Verbesserte Unterstützung für die lokale ETL-Skriptentwicklung mithilfe der öffentlichen AWS Glue -ETL-Bibliothek	Der Inhalt der AWS Glue ETL-Bibliothek wurde aktualisiert, um widerzuspiegeln, dass AWS Glue Version 1.0 jetzt unterstützt wird. Weitere Informationen finden Sie unter Lokales Entwickeln und Testen von ETL-Skripts mithilfe der AWS Glue ETL-Bibliothek.	18. September 2019
Unterstützung für das Ausschließen von Amazon-S3-Speicherklassen bei der Ausführung von Aufträgen	Es wurden Informationen zum Ausschluss von Amazon S3-Speicherklassen bei der Ausführung von AWS Glue ETL-Jobs hinzugefügt, die Dateien oder Partitionen aus Amazon S3 lesen. Weitere Informationen finden Sie unter Ausschließen von Amazon-S3-Speicherklassen.	29. August 2019
Support für die lokale ETL-Skriptentwicklung mithilfe der öffentlichen AWS Glue ETL-Bibliothek	Es wurden Informationen zum lokalen Entwickeln und Testen von Python und Scala ETL-Skripts hinzugefügt, ohne dass eine Netzwerkverbindung erforderlich ist. Weitere Informationen finden Sie unter Lokales Entwickeln und Testen von ETL-Skripts mithilfe der AWS Glue ETL-Bibliothek.	28. August 2019

Bekannte Probleme	Es wurden Informationen zu bekannten Problemen in AWS Glue hinzugefügt. Weitere Informationen finden Sie unter Bekannte Probleme für AWS Glue.	28. August 2019
Die Support für maschinelles Lernen transformiert sich in AWS Glue	Es wurden Informationen zu Funktionen für maschinelles Lernen hinzugefügt, die von AWS Glue zur Erstellung benutzerdefinierter Transformationen bereitgestellt werden. Sie können diese Transformationen erstellen, wenn Sie einen Auftrag erstellen. Weitere Informationen finden Sie unter Transformationen durch Machine Learning in AWS Glue.	8. August 2019
Unterstützung für gemeinsam genutzte Amazon Virtual Private Clouds	Es wurden Informationen zur AWS Glue Unterstützung der gemeinsam genutzten Amazon Virtual Private Cloud hinzugefügt. Weitere Informationen finden Sie unter Shared Amazon VPCs.	6. August 2019

Support für Versionierung in AWS Glue	Es wurden Informationen zur Definition der `Glue version` In-Job-Eigenschaften hinzugefügt. AWS Glue Version bestimmt die Versionen von Apache Spark und Python, die AWS Glue unterstützt werden. Weitere Informationen finden Sie unter Jobs hinzufügen in AWS Glue.	24. Juli 2019
Unterstützung für zusätzliche Konfigurationsoptionen für Entwicklungsendpunkte	Es wurden Informationen zu Konfigurationsoptionen für Entwicklungsendpunkte mit speicherintensiven Workloads hinzugefügt. Sie haben die Wahl zwischen zwei neuen Konfigurationen, die mehr Speicher pro Executor bieten. Weitere Informationen finden Sie unter Arbeiten mit Entwicklungsendpunkten auf der AWS Glue Konsole.	24. Juli 2019

Unterstützung von Aktivitäten zum Extrahieren, Transformieren und Laden von Daten (ETL) mithilfe von Workflows	Es wurden Informationen zur Verwendung eines neuen Konstrukts hinzugefügt, das als Workflow bezeichnet wird, um eine komplexe ETL-Aktivität (Extrahieren, Transformieren und Laden) mit mehreren Jobs zu entwerfen, die als eine Einheit ausgeführt und verfolgt werden AWS Glue kann. Weitere Informationen finden Sie unter Durchführen komplexer ETL-Aktivitäten mithilfe von Workflows in AWS Glue.	20. Juni 2019
Unterstützung für Python 3.6 in Python-Shell-Aufträgen	Informationen zur Unterstützung für Python 3.6 in Python-Shell-Aufträgen hinzugefügt. Sie können entweder Python 2.7 oder Python 3.6 als Auftragseigenschaft angeben. Weitere Informationen finden Sie unter Hinzufügen von Python-Shell-Aufträgen in AWS Glue.	5. Juni 2019

Unterstützung für Virtual Private Cloud (VPC)-Endpunkte	Es wurden Informationen zur direkten Verbindung AWS Glue über einen Schnittstellenendpunkt in Ihrer VPC hinzugefügt. Wenn Sie einen VPC-Schnittstellenendpunkt verwenden, AWS Glue erfolgt die Kommunikation zwischen Ihrer VPC und der VPC vollständig und sicher innerhalb des AWS Netzwerks. Weitere Informationen finden Sie unter Verwenden AWS Glue mit VPC-Endpunkten.	4. Juni 2019
Support für die kontinuierliche Protokollierung von AWS Glue Jobs in Echtzeit.	Es wurden Informationen zur Aktivierung und Anzeige von Apache Spark-Jobprotokollen in Echtzeit hinzugefügt, CloudWatch einschließlich der Treiberprotokolle, der einzelnen Executor-Protokolle und eines Spark-Job-Fortschrittsbalkens. Weitere Informationen finden Sie unter Continuous Logging for AWS Glue Jobs.	28. Mai 2019

Unterstützung für vorhandene Data-Catalog-Tabellen als Crawler-Quellen	Es wurden Informationen zum Angeben einer Liste von vorhandenen Data-Catalog-Tabellen als Crawler-Quellen hinzugefügt. Crawler können dann Änderungen an Tabellen-Schemata erkennen, Tabellendefinitionen aktualisieren und neue Partitionen registrieren, wenn neue Daten verfügbar werden. Weitere Informationen finden Sie unter Crawler-Eigenschaften.	10. Mai 2019
Unterstützung für zusätzliche Konfigurationsoptionen für speicherintensive Aufträge	Zusätzliche Informationen zu den Konfigurationsoptionen für Apache-Spark-Aufgaben mit speicherintensiven Workloads. Sie haben die Wahl zwischen zwei neuen Konfigurationen, die mehr Speicher pro Executor bieten. Weitere Informationen finden Sie unter Jobs hinzufügen in. AWS Glue	5. April 2019
Unterstützung für benutzerdefinierte CSV-Classifier	Zusätzliche Informationen zur Verwendung eines benutzerdefinierten CSV-Classifier zum Ableiten des Schemas verschiedener Typen von CSV-Daten. Weitere Informationen finden Sie unter Schreiben benutzerdefinierter Classifier.	26. März 2019

Support für AWS Ressourcen-Tags	Es wurden Informationen zur Verwendung von AWS Ressourcen-Tags hinzugefügt, mit denen Sie den Zugriff auf Ihre AWS Glue Ressourcen verwalten und kontrollieren können. In können Sie Jobs, Triggern, Endpunkten und Crawlern AWS Ressourcen-Tags zuweisen. AWS Glue Weitere Informationen finden Sie unter AWS Tags in. AWS Glue	20. März 2019
Unterstützung von Data Catalog für Spark-SQL-Aufträge	Es wurden Informationen zur Konfiguration Ihrer AWS Glue Jobs und Entwicklungsendpunkte hinzugefügt, um den AWS Glue Data Catalog als externen Apache Hive Metastore zu verwenden. Auf diese Weise können Aufträge und Entwicklungsendpunkte direkt Apache-Spark-SQL-Abfragen für im AWS Glue Data Catalog gespeicherte Tabellen ausführen. Weitere Informationen finden Sie unter AWS Glue Data Catalog -Unterstützung für Spark-SQL-Aufträge.	14. März 2019

Unterstützung für Python-Shell-Aufträge	Hinzufügung von Informationen zu Python-Shell-Aufträgen und zum neuen Feld Maximum capacity (Maximale Kapazität). Weitere Informationen finden Sie unter Hinzufügen von Python-Shell-Aufträgen in AWS Glue.	18. Januar 2019
Unterstützung für Benachrichtigungen zu Änderungen bei Datenbanken und Tabellen	Hinzufügung von Informationen zu Ereignissen, die bei Änderungen von Datenbank-, Tabellen- und Partitions-API-Aufrufen generiert werden. Sie können unter Ereignisse Aktionen konfigurieren, um auf diese CloudWatch Ereignisse zu reagieren. Weitere Informationen finden Sie unter Automatisieren AWS Glue mit CloudWatch Ereignissen.	16. Januar 2019
Unterstützung für die Verschlüsselung von Verbindungspasswörtern	Es wurden zusätzliche Informationen zum Verschlüsseln von Passwörtern, die in Verbindungsobjekten verwendet werden, hinzugefügt. Weitere Informationen finden Sie unter Verbindungspasswörter.	11. Dezember 2018

Unterstützung für Berechtigungen auf Ressourcenebene und ressourcenbasierte Richtlinien	Es wurden Informationen zur Verwendung von Berechtigungen auf Ressourcenebene und ressourcenbasierten Richtlinien mit hinzugefügt. AWS Glue Weitere Informationen finden Sie in den Themen unter Sicherheit in AWS Glue.	15. Oktober 2018
Support für SageMaker KI-Notebooks	Es wurden Informationen zur Verwendung von SageMaker KI-Notebooks mit AWS Glue Entwicklungsendpunkten hinzugefügt. Weitere Informationen finden Sie unter Verwalten von Notebooks.	5. Oktober 2018
Unterstützung für Verschlüsselung	Es wurden Informationen zur Verwendung von Verschlüsselung mit AWS Glue hinzugefügt. Weitere Informationen finden Sie unter Verschlüsselung im Ruhezustand, Verschlüsselung während der Übertragung und Einrichten der Verschlüsselung in AWS Glue.	24. August 2018

Unterstützung für Apache-Spark-Auftragsmetriken	Informationen über die Verwendung von Apache-Spark-Metriken zum besseren Debuggen und Profilieren von ETL-Aufträgen hinzugefügt. Von der Konsole aus können Sie auf einfache Weise Laufzeitmetriken wie gelesene und geschriebene Byte, Speicherauslastung und CPU-Last des Treibers und der Executoren sowie Datenaustausch zwischen Executoren verfolgen. AWS Glue Weitere Informationen finden Sie unter Überwachung AWS Glue mithilfe von CloudWatch Metriken, Job-Überwachung und Debugging und Arbeiten mit Jobs auf der AWS Glue Konsole.	13. Juli 2018
Unterstützung von DynamoDB als Datenquelle	Zusätzliche Informationen zum Crawling von DynamoDB und ihre Verwendung als Datenquelle von ETL-Aufträgen. Weitere Informationen finden Sie unter Katalogisieren von Tabellen mit einem Crawler und Verbindungsparameter.	10. Juli 2018

Aktualisierungen zum Verfahren für das Erstellen des Notebook-Servers	Aktualisierte Informationen zum Erstellen eines Notebook-Servers auf einer EC2 Amazon-Instance, die mit einem Entwicklungsendpunkt verknüpft ist. Weitere Informationen finden Sie unter Erstellen eines Notebook-Servers, der einem Entwicklungsendpunkt zugeordnet ist.	9. Juli 2018
Updates jetzt über RSS verfügbar	Sie können jetzt einen RSS-Feed abonnieren, um Benachrichtigungen über Aktualisierungen im AWS Glue -Entwicklerhandbuch zu erhalten.	25. Juni 2018
Unterstützung für Benachrichtigungen zu Auftragsverzögerungen	Informationen zum Konfigurieren eines Verzögerungsschwellenwerts beim Ausführen eines Auftrags hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Aufträgen in AWS Glue.	25. Mai 2018
Konfigurieren eines Crawlers zum Anhängen neuer Spalten	Es wurden Informationen zur neuen Konfigurationsoption für Crawler hinzugefügt, MergeNewColumns. Weitere Informationen finden Sie unter Konfigurieren eines Crawlers.	7. Mai 2018

Unterstützung der Zeitüberschreitung bei Aufträgen	Informationen zum Einrichten eines Timeout-Schwellenwerts beim Ausführen eines Auftrags hinzugefügt. Weitere Informationen finden Sie unter Hinzufügen von Aufträgen in AWS Glue.	10. April 2018
Unterstützung von Scala-ETL-Skripts und Auslösen von Aufträgen basierend auf zusätzlichen Ausführungszuständen	Informationen zur Verwendung von Scala als ETL-Programmiersprache wurden hinzugefügt. Darüber hinaus unterstützt die Auslöser-API jetzt das Auslösen, wenn beliebige Bedingungen erfüllt sind (zusätzlich zu allen Bedingungen). Außerdem können Aufträge basierend auf einer "fehlerhaften" oder "angehaltenen" Auftragsausführung ausgelöst werden (zusätzlich zu einer "erfolgreichen" Auftragsausführung).	12. Januar 2018

Frühere Aktualisierungen

In der folgenden Tabelle sind die wichtigen Änderungen in jeder Version des AWS Glue - Entwicklerhandbuchs vor Januar 2018 beschrieben.

Änderung	Beschreibung	Datum
Support von XML-Datenquellen und eine neue Option für die Crawler-Konfiguration	Informationen zur Klassifizierung von XML-Datenquellen und einer neuen Crawler-Option für Partitionsänderungen wurden hinzugefügt.	16. November 2017

Änderung	Beschreibung	Datum
Neue Transformationen, Unterstützung für zusätzliche Amazon-RDS-Datenbank-Engines und Erweiterungen für Entwicklungsendpunkte	Informationen zu den Transformationen Zuordnung und Filter, Unterstützung für Amazon RDS Microsoft SQL Server und Amazon RDS Oracle und neuen Features für Entwicklungsendpunkte wurden hinzugefügt.	29. September 2017
AWS Glue erste Veröffentlichung	Dies ist die erste Version des AWS Glue - Entwicklerhandbuchs.	14. August 2017

AWS Glossar

Die neueste AWS Terminologie finden Sie im AWS Glossar in der AWS-Glossar Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.